VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ
BRNO UNIVERSITY OF TECHNOLOGY

FAKULTA INFORMAČNÍCH TECHNOLOGIÍ
FACULTY OF INFORMATION TECHNOLOGY

ÚSTAV POČÍTAČOVÉ GRAFIKY A MULTIMÉDIÍ
DEPARTMENT OF COMPUTER GRAPHICS AND MULTIMEDIA

OVLÁDÁNÍ HUDEBNÍHO „COMBA“

CONTROL OF MUSICAL "COMBO"

BAKALÁŘSKÁ PRÁCE
BACHELOR'S THESIS

AUTOR PRÁCE — **FRANTIŠEK JEŘÁBEK**
AUTHOR

VEDOUCÍ PRÁCE — **prof. Dr. Ing. PAVEL ZEMČÍK**
SUPERVISOR

BRNO 2021

Vysoké učení technické v Brně
Fakulta informačních technologií

Ústav počítačové grafiky a multimédií (UPGM) Akademický rok 2020/2021

Zadání bakalářské práce

Student: **Jeřábek František**
Program: Informační technologie
Název: **Ovládání hudebního "Comba"**
Control of Musical "Combo"
Kategorie: Zpracování signálů

Zadání:

1. Seznamte se s principy komunikace embedded systémů s hudebními systémy typu "Combo" (například pro kytaru) a s možnostmi komunikace a protokoly pro komunikaci.
2. Vyberte vhodný embedded systém a vhodný "Combo" systém, navrhněte postup pro jejich komunikaci a také vhodné uživatelské rozhraní například pro nastavování parametrů a také pro ovládání systému při hře na hudební nástroj.
3. Popište možnosti a vlastnosti navrženého řešení a diskutujte možnosti jeho implementace.
4. Implementujte navržené řešení a demonstrujte jeho vlastnosti na vhodném příkladu "Comba" a hudebního nástroje.
5. Diskutujte dosažené vlastnosti a možnosti dalšího rozvoje práce.

Literatura:

- Podle instrukcí vedoucího práce

Pro udělení zápočtu za první semestr je požadováno:

- Body 1 až 3 zadání.

Podrobné závazné pokyny pro vypracování práce viz https://www.fit.vut.cz/study/theses/

Vedoucí práce: **Zemčík Pavel, prof. Dr. Ing.**
Vedoucí ústavu: Černocký Jan, doc. Dr. Ing.
Datum zadání: 1. listopadu 2020
Datum odevzdání: 12. května 2021
Datum schválení: 30. října 2020

Abstrakt

Tato práce se zabývá tvorbou pedálového ovladače pro kytarové kombo Yamaha THR10 za pomocí mikropočítače Raspberry Pi Zero W a vývojem mobilní aplikace pro uživatelskou interakci s ovladačem. Je zde popsán postup analýzy a implementace předem neznámého komunikačního protokolu komba a proprietární počítačové aplikace. Práce popisuje tvorbu komunikačního protokolu interních součástí ovladače a textového protokolu ve formátu JSON pro interakci ovladače a mobilní aplikace.

Abstract

This thesis focuses on creating a pedal controller for guitar combo Yamaha THR10 using Raspberry Pi Zero W microcomputer and a mobile application for configuration of the controller. It describes analysis and implementation of previously unknown protocol used for communication between guitar combo and configuration application. It also describes the creation of custom protocol used in internal communication and textual protocol in JSON for communication between pedal controller and mobile application

Klíčová slova

Yamaha THR10, Raspberry Pi Zero W, Android, MIDI, JSON, Mobilní aplikace, kytarový pedál, komunikační protokol

Keywords

Yamaha THR10, Raspberry Pi Zero W, Android, MIDI, JSON, Mobile app, guitar pedal, communication protocol

Citace

JEŘÁBEK, František. *Ovládání hudebního „Comba“*. Brno, 2021. Bakalářská práce. Vysoké učení technické v Brně, Fakulta informačních technologií. Vedoucí práce prof. Dr. Ing. Pavel Zemčík

Ovládání hudebního „Comba“

Prohlášení

Prohlašuji, že jsem tuto bakalářskou práci vypracoval samostatně pod vedením pana prof. Dr. Ing. Pavla Zemčíka. Uvedl jsem všechny literární prameny, publikace a další zdroje, ze kterých jsem čerpal.

.....
František Jeřábek
9. května 2021

Poděkování

Chtěl bych poděkovat panu prof. Dr. Ing. Pavlu Zemčíkovi za odborné vedení práce, konzultace a trpělivost. Dále bych chtěl poděkovat Ing. Ondřeji Jeřábkovi za tvorbu rozšiřující desky hardwaru.

Obsah

Kapitola 1

Úvod

V dnešní době se ve velkém množství hudebních skladeb mění mnohokrát tón elektrické kytary. Tyto změny tónu obstarává v hudebních skupinách zvukař. Pokud se jednotlivec snaží tyto skladby hrát, nastává problém, kdy je složité tuto změnu tónu provést. Mnoho kytarových zesilovačů tuto funkci poskytuje pomocí pedálových ovladačů, které umožňují měnit mezi předem připravenými nastaveními. Velké množství kytarových komb (kombinace kytarového zesilovače a reproduktoru) tuto možnost neposkytuje. Některá z nich však umožňují nastavení kytarového tónu, efektů a hlasitosti pomocí počítačové aplikace.

Cílem této práce je vytvořit ovládací pedál, s použitím již existujícího komunikačního protokolu, který je použit pro nastavování komba pomocí počítačové aplikace. Navrhnutí mobilní aplikace s přehledným uživatelským prostředím. Ta by sloužila jako náhrada počítačové aplikace a zároveň by ji rozšiřovala o funkce pro nastavení a zobrazování informací z ovládacího pedálu. Je nutností, aby byl ovladač funkční i bez připojení k mobilní aplikaci.

Pro toto téma jsem se rozhodl, z důvodu snahy o co nejpřesnější hru hudebních skladeb na elektrickou kytaru i se změnami kytarového tónu v průběhu skladby. Tento problém by bylo možné vyřešit pořízením kytarového komba, umožňujícího přepínání tónů. Takové typy mohou být příliš drahé. Jelikož kytarové kombo, pro které je tento ovladač vytvářen, umožňuje přehrávat a nahrávat zvuk bude možné aktuální práci v budoucnu rozšířit i o tyto funkce.

Tato práce je rozdělena do čtyř kapitol. První z nich popisuje mnou vybrané existující produkty a shrnuje různé postupy, kterými je umožněno uživateli měnit kytarový tón. V druhé kapitole jsou popsány současné technologie použitelné pro tvorbu pedálového ovladače. Najdete zde shrnutí používaných komunikačních technologií, ale také principy použité pro tvorbu uživatelského prostředí mobilních aplikací. Kapitola Analýza stavu a návrh řešení obsahuje souhrn a zhodnocení aktuálních technologií, blíže specifikuje zadání práce a rozděluje problém do menších, snadněji pochopitelnějších, celků. Poslední kapitola popisuje samotné řešení jednotlivých problémů specifikovaných v předchozí kapitole a spojení jednotlivých celků do konečného stavu. Na konci této kapitoly je také popis testování kompletního systému.

Kapitola 2

Existující řešení ovladačů pro kytarová komba

V této kapitole jsou popsána existující řešení ovladačů kytarového tónu. Existuje nepřeberné množství kytarových pedálů, které modifikují tón. Tyto kytarové pedály se většinou připojují do efektové smyčky použitého zesilovače. Všechny zesilovače ale nedisponují touto efektovou smyčkou a při připojení pedálu před zesilovač, nemusí dosahovat požadovaných výsledků.

Cílem této kapitoly je shrnout možnosti, které poskytují již existující ovladače kytarových komb. Jsou zde uvedeny informace související s bakalářskou prací. Nejedná se o encyklopedický přehled s ohledem na omezení rozsahu bakalářské práce.

2.1 Yamaha THR10 II

Toto kombo je novější verze kytarového komba použitého v této práci. Umožňuje bezdrátové nastavení tónu pomocí mobilní aplikace a technologie Bluetooth[9]. Kombo také dokáže bezdrátově přehrávat a nahrávat audio. Bezdrátově je také možné připojit samotný hudební nástroj, díky vestavěnému přijímači, který je kompatibilní s vysílačem *Line 6 G10T*[9].

Obrázek 2.1: Kytarové kombo THR10-II.

Ke konfiguraci komba je možné použít mobilní zařízení, které lze po instalaci aplikace *THR Remote* propojit s hudebním kombem. Aplikace je dostupná ke stažení z Google Play, nebo App store [9]. Mohou zde být nastaveny všechny hodnoty, které se dají nastavit pomocí

fyzických knoflíků na kombu. Aplikace také umožňuje ukládání nastavení do paměti komba. Při uložení nastavení z aplikace do jedné z pěti pamětí je možné již bez použití mobilní aplikace mezi nimi přepínat. Spolu s tímto ovladačem je možné použít pedálové přepínače, které využívají bezdrátovou technologii Bluetooth. Mobilní aplikace má přívětivé uživatelské rozhraní a umožňuje detailní nastavení možností komba.

Obrázek 2.2: Mobilní aplikace THR remote.

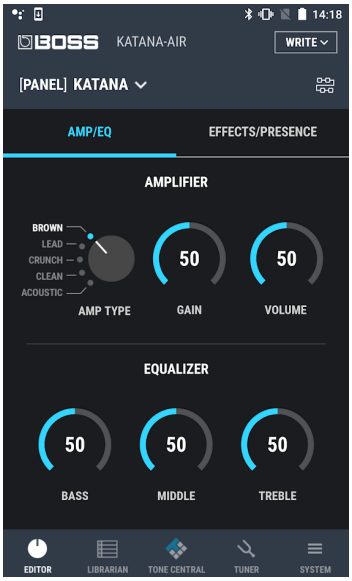
2.2 BOSS Katana-Air

Jedná se o kytarové kombo, které umožňuje kompletně bezdrátový provoz. Hudební nástroj je ke kombu připojen pomocí vysílače připojeného přímo do hudebního nástroje. Kytarové kombo má vlastní vestavěný přijímač. BOSS Katana-Air dosahuje výkonu 20W pokud je napájen pomocí baterií, nebo 30W pokud je napájen pomocí adaptéru ze sítě [3].

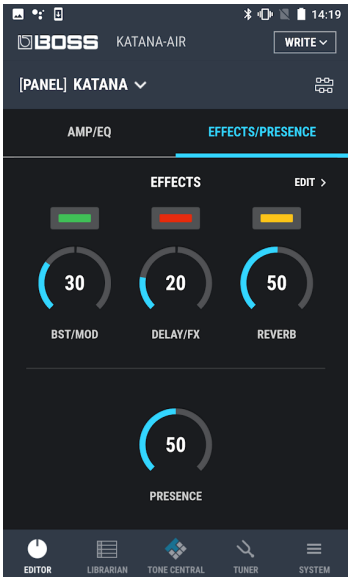
Vysílač, který umožňuje připojení hudebního nástroje má vlastní vestavěnou nabíjecí baterii, na kterou je schopen pracovat po dobu 12-ti hodin [3]. Tuto baterii je možné nabíjet vložením do samotného komba.

Obrázek 2.3: Kytarové kombo BOSS-katana-Air.

Kombo lze také konfigurovat pomocí mobilní aplikace. Ta je připojena pomocí technologie Bluetooth za pomoci mobilní aplikace *BTS for KATANA-AIR*. V aplikaci je po připojení možné nastavovat základní nastavení, jako například typ zesilovače, hlasitost a ekvalizér. Po přepnutí do záložky *EFFECTS/PRESENCE* je možné konfigurovat efekty. Aplikace také umožňuje výběr z předem vytvořených nastavení [5].



Obrázek 2.4: Základní nastavení.



Obrázek 2.5: Nastavení efektů.

2.3 MARSHALL DSL40CR

Jedná se o lampové kytarové kombo s výkonem 40W [4] se dvěma kanály. Umožňuje pomocí ovladače *Gain* navolit od čistého kytarového tónu až po zkreslený „metalový“. Kombo také disponuje efektovou smyčkou, pomocí které je možné dále modifikovat kytarový tón. Jednotlivé kanály umožňují samostatné nastavení hodnot „Gain“, „Reverb“, „Master Volume“, a také nastavení dvou zvukových režimů. Oba kanály sdílí třípásmový ekvalizér. Na zadní straně komba se nachází výstup do externích reproduktorů či „audio in“ vstup pro připojení externího zvukového zdroje. Ke kombu lze připojit pedálový přepínač, který umožňuje přepínání mezi jednotlivými kanály, případně aktivaci efektové smyčky. Taktéž se prodává šestikanálový pedálový přepínač, který poskytuje rozšířené možnosti ovládání.

Obrázek 2.6: Kytarové kombo Marshall DSL40CR.

2.4 Line 6 HX Stomp

Line 6 HX Stomp je multiefektový pedál s možností modifikace kytarového tónu pomocí 200 efektů, emulaci 74 zesilovačů a 37 kabinetů a mikrofonů. Toto zařízení také přináší nadstandardní dynamické rozpětí 123dB [7]. Lze použit současně až 6 efektových bloků, které obsahují mnoho možností konfigurace. Efekty je možné řadit za sebe v libovolném pořadí a vytvářet tak množství unikátních tónů. Takto vytvořené nastavení je možné uložit do jedné ze 3 pamětí, mezi kterými je možné instantně přepínat i v průběhu skladby.

Veškeré nastavení lze provádět přímo na zařízení skrze LCD displej (320 x 240 pixelů) a pomocí tlačítek. Pedálové přepínače můžou být nastaveny na ovládání specifického efektu nebo k přepínání uložených nastavení či úplnému vyřazení efektového systému. Line 6 HX Stomp obsahuje vestavěný ekvalizér, který umožňuje další modifikace tónu.

Toto zařízení také obsahuje vestavěnou ladičku s množstvím nastavení jako je například ladění v nestandardních intervalech. Lze využít i funkci „loop pedálu“, která je zde

také obsažena. Při použití funkce emulace QWERTY klávesnice je možné tento pedál použít k ovládání většiny počítačového softwaru. Funkce odesílaní midi poskytuje ovládání kompatibilních zařízeni jako například osvětlení.

Obrázek 2.7: Multiefektový pedál Line 6 HX stomp

Konfigurační aplikace Pro Line 6 HX Stomp není k dispozici mobilní konfigurační aplikace. K dispozici je aplikace pro PC. Ta umožňuje po připojení pomocí USB pohodlně provádět konfigurace. Ty jsou poté nahrány a uloženy v zařízení a je možné je použít i bez připojení k PC. Aplikace také umožňuje vytvářet a obnovovat zálohy zařízení nebo dokonce stahování již předem připravených nastavení.

Kapitola 3

Současné technologie pro tvorbu pedálového ovladače

K vytvoření vlastního pedálového ovladače je nutné provést analýzu aktuálních technologií. Tato analýza usnadní rozdělení tvorby pedálu do podproblémů a umožní určit postupy použité při jejich implementaci. Také zjednoduší výběr komunikačních protokolů a standardů použitých k jejich propojení.

V této kapitole jsou popsány technologie relevantní k řešenému problému. Nejedná se o encyklopedii a vzhledem k rozsahu práce zde nejsou popsány všechny technologie, použitelné pro řešení jednotlivých problémů, kterých je velké množství. Příkladem může být komunikační protokol pro komunikaci se samotným kombem. Zde bude popsán pouze protokol MIDI, který je typický pro komunikaci se zvukovými zařízeními. Také zde nejsou popsány všeobecně známé technologie pro osobu znalou základů informačních technologií jako je například JSON či YAML.

3.1 Raspberry Pi Zero W

Raspberry PI zero W je jednodeskový počítač (SBC) obsahující jednojádrový procesor *ARMv6* o taktu 1GHz. Tento jednodeskový počítač rozšiřuje rodinu Raspberry Pi Zero. Má veškerou funkcionalitu originálního Pi Zero a doplňuje jeho funkcionalitu o bezdrátovou konektivitu:

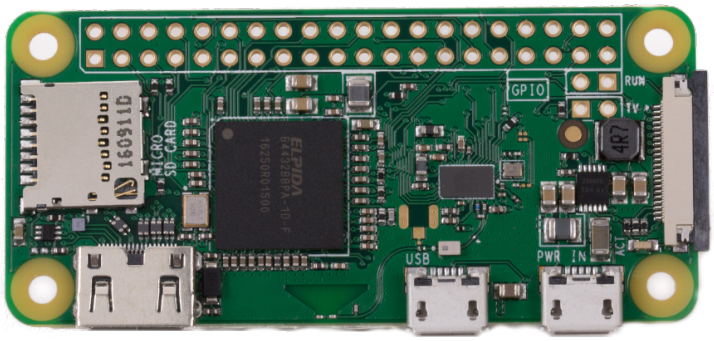
- 802.11 b/g/n wireless LAN
- Bluetooth 4.1
- Bluetooth Low Energy (BLE)

Stejně jako *Pi Zero* má i *Pi Zero W*:

- 1GHz jedno-jádrový procesor
- 512MB RAM
- Mini HDMI port
- Micro USP OTG port
- Micro USB Power

- HAT-kompatibilní 40-pinový konektor

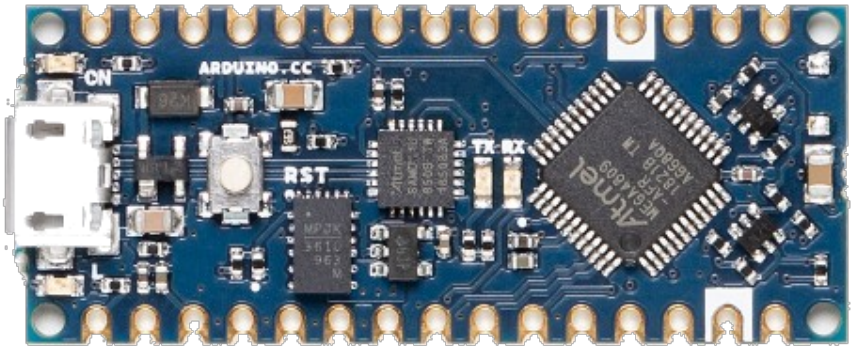
Raspberry Pi Zero W podporuje operační systém linux, který následně umožňuje komunikaci s připojenými periferiemi. Tento jednodeskový počítač je relativně malý s velikostí: *65mm x 30mm x 5mm* [16].



Obrázek 3.1: Raspbery Pi Zero W.

3.2 Arduino Nano Every

Arduino Nano Every je vývojová platforma kompatibilní s 5V logikou. Na desce je obsažen procesor *ATMega4809*, který má flash paměť 48KB, SRAM 6KB a EEPROM 256 bytů [6]. Procesor je taktován na 20MHz. Vývojová platforma obsahuje jedno rozhraní UART, SPI a I2C. Arduino Nano Every má rozměry *45mm x 18mm* .



Obrázek 3.2: Arduino Nano Every.

3.3 Komunikace po lokální síti

Komunikaci po lokální síti je možné realizovat pomocí UNIX socketů. Sockety umožňují komunikaci několika zařízení/programů mezi sebou za použití unixových „file descriptorů“. Existuje několik typů unixových socketů. Dva hlavní typy jsou proudové sockety „SOCK_STREAM“ a datagramové sockety „SOCK_DGRAM“ [17].

Proudové sockety Jedná se o spolehlivé, obousměrné, spojované proudy. Zachovávají pořadí odeslaných zpráv. Tyto sockety používají pro komunikaci protokol TCP [17]. Každá odeslaná zpráva je potvrzována a pokud není potvrzena dochází k opakovanému odeslání této zprávy.

Datagramové sockety Datagramové sockety jsou nespojované a nespolehlivé. Není zaručeno, že odeslané zprávy budou přijaty ve stejném pořadí jako byly odeslány. Není také zaručeno, že zprávy budou přijaty. Pro odesílání pomocí tohoto typu socketu je použit protokol UDP [17]. Tento typ je rychlejší než proudová varianta, jelikož odeslané zprávy nejsou potvrzovány a není tedy třeba čekat na potvrzení.

Číslo portu Každá aplikace, která komunikuje po síti, má přiřazené číslo portu, pomocí kterého komunikuje s ostatními zařízeními. Tímto je možné odlišit více aplikací používajících síťovou komunikaci na jednom zařízení [17].

3.4 Technologie Bluetooth

Jedná se o technologii umožňující ad hoc bezdrátovou komunikaci na malé vzdálenosti s nízkou spotřebou a cenou. Technologie Bluetooth umožňuje propojení více zařízení v relativní blízkosti po stejném přenosovém médiu bez nutnosti koordinace. Přenos probíhá pomocí rádiových vln na frekvencích přibližně 2,45GHz [15] .

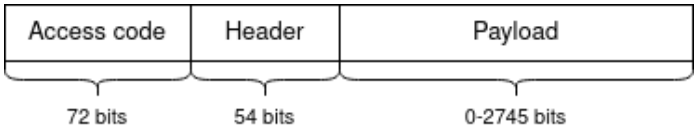
Vzhledem k nepředvídatelnosti množství zařízení vysílajících na této frekvenci, zajišťuje Bluetooth odolnost proti rušení pomocí filtrace ostatních frekvencí. Zařízení propojené touto technologií mohou být připojené k více zařízením současně. Tato funkcionalita je zajištěna modulací pomocí technologie FH-CDMA [1] [15].

Pro zajištění obousměrného spojení je použito TDD [2], které pracuje na základě rozdělení komunikace na časové intervaly(sloty). Ty jsou přiděleny jednotlivým účastníkům pro vysílání/přijímání. Komunikace je fragmentována do paketů. V každém komunikačním slotu je možné odeslat pouze jeden paket.

Všechny pakety mají stejnou strukturu znázorněnou na obrázku 3.3. Hodnota „Access code“ udává kód pro rozlišení komunikace po jednom kanálu. Příjemce kontroluje tento kód s očekávaným kódem a přijme jej pouze pokud se shodují. Hlavička paketu obsahuje adresu o délce 3 bity. Dále také obsahuje 1 bit pro potvrzení/odmítnutí zprávy pro automatický požadavek o znovu-odeslání ARQ[3], 4 bity pro označení typu zprávy, 8 bitů pro detekci chyb (CRC) [15].

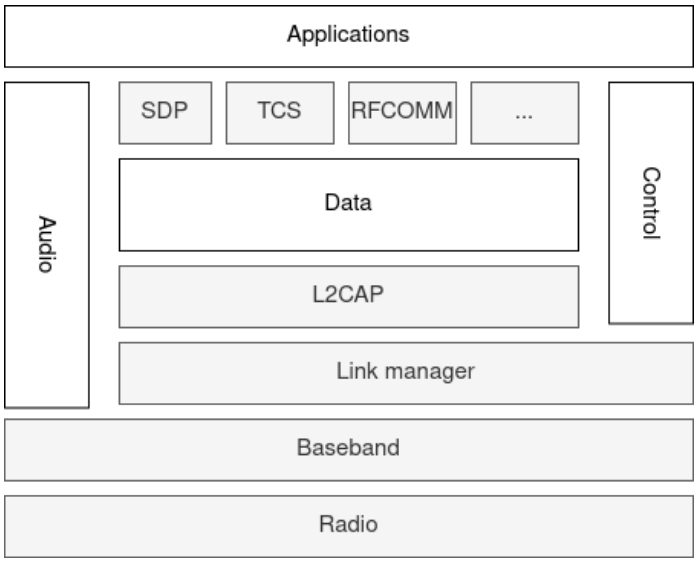
Bluetooth specifikuje několik protokolů, které jsou součástí tzv. „Bluetooth protocol stack“ znázorněný na obrázku 3.4. Ten je složen z na sobě závisejících vrsev. Protokoly niž-

[1] Frequency hopping code division multiple access [12].
[2] Time division duplexing
[3] Automatic repeat request



Obrázek 3.3: Struktura bluetooth packetu [15].

ších vrstev poskytují rozhraní vyšším vrstvám, které umožňují vyšší abstrakci nad celkovou komunikací.



Obrázek 3.4: Struktura "Bluetooth protocol stack".

Protokol L2CAP

Logical Link Controll and Adaptation Protocol zprostředkovává spojovanou i nespojovanou komunikaci protokolům vyšších vrstev. Podporuje segmentaci a zpětnou rekonstrukci dat [13]. Také umožňuje vyšším vrstvám odesílat a přijímat pakety nižších vrstev. Provádí "Flow control"a obsluhuje znovu-odesílání ztracených (nebo chybně přijatých) paketů (ARQ)[4].

L2CAP používá kanály což je logické spojení dvou zařízení po jednom přenosovém spojení, ty mají přiřazené označení pomocí CID (channel identifier). CID udává unikátní lokální označení pro kanály ve spojení k jinému zařízení. Kanály *0x0001* - *0x003F* jsou rezervovány pro specifické funkce protokolu. Kanál s označením *0x0001* slouží k vytvoření a správě spojovaných připojení. Posílají se zde charakteristiky spojení a jejich případné změny. Kanál *0x0002* slouží pro příchozí a odchozí data při nespojované komunikaci [13].

Každý kanál může pracovat v jednom z několika operačních módů:

- Základní L2CAP
- Flow Control
- Retransmisison

[4] Automatic repeat request

- Enhanced Retransmission
- Streaming
- LE Credit Based Flow Control
- Enhnaced Credit Based Flow Control

Základní L2CAP mód je použit pokud se při spojení zařízení nedohodly na jiném. V módech Flow Control, Retransmission a Enhanced Retransmission jsou odesílané/přijímané pakety očíslovány a potvrzovány. Číslování je použito pro vyrovnávací paměť. *Flow Control* neprovádí znovu odeslání ztracených paketů, ale umožňuje jejich detekci a identifikaci.

Retransmission používá časovač pro zaručení doručení paketu. Pokud není přijata potvrzovací zpráva je provedeno znovu odesláni paketu za použití algoritmu *Go-Back-N*.

Enhanced Retransmission je podobný jako Retransmission mód. Zvyšuje efektivitu zotavení z chyb. Přidává RNR paket pro označení zaneprázdněnosti zařízení.

Streaming mód slouží pro real-time izochronní komunikaci. Pakety jsou číslované, ale nejsou potvrzované. Umožňuje detekci ztracených paketů

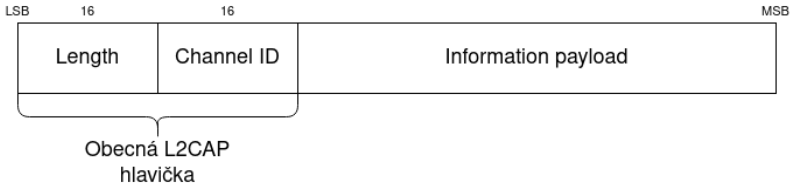
LE Credit Based Flow Control je použit pro LE spojovanou komunikaci pomocí L2CAP za použití „Credit based Flow Control“ [13]. Pakety jsou odesílány dokud má příjemce dostatečný „kredit“. Stav kreditu je odesílán při potvrzování přijatých dat.

Enhanced Credit Based Flow Control je podobný jako „LE Credit Based Flow Control“, ale je možné tento mód použít nejen pro LE [13].

Každý mód používá specifickou strukturu paketu. Všechny pakety obsahují společnou hlavičku, za kterou následují hodnoty specifické pro daný mód.

Základní L2CAP spojovaný mód L2CAP ve spojovaném módu má strukturu paketu znázorněnou na obrázku 3.5.

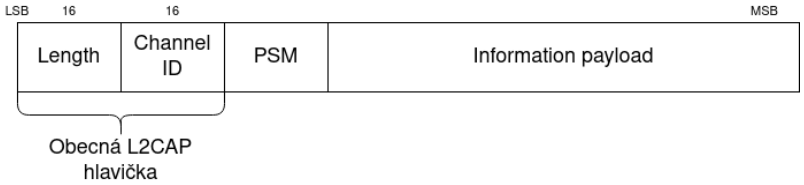
- Length je hodnota o velikosti 2B, která udává velikost paketu bez úvodní hlavičky. Maximální velikost je 65535.
- Channel ID o velikosti 2B. Udává identifikátor (CID) cílového kanálu, na který je paket doručen.
- Information payload obsahuje samotná data obdržená z vyšších protokolů při odesílání nebo předávaná vyšším protokolům po přijetí paketu.



Obrázek 3.5: Struktura paketu v základním L2CAP spojovaném módu [13].

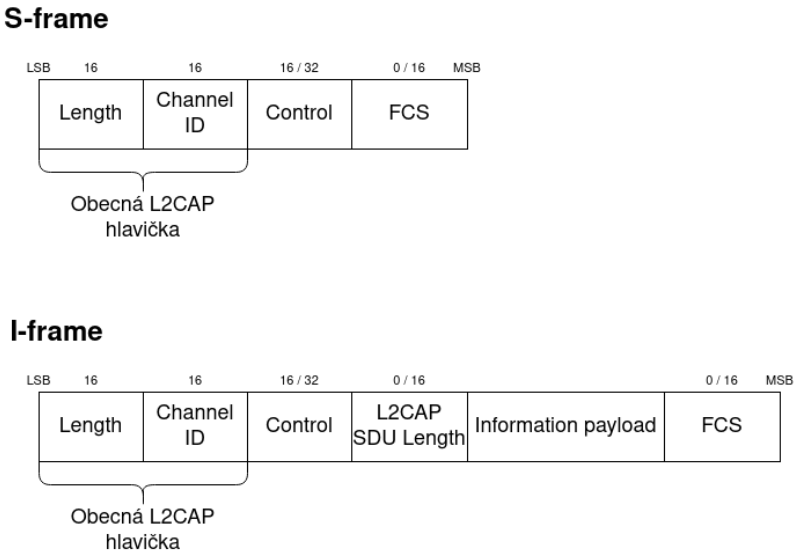
Základní L2CAP nespojovaný mód L2CAP v nespojovaném módu má strukturu paketu znázorněnou na obrázku 3.6

- Length (2B) Udává velikost paketu bez úvodní hlavičky plus velikost hodnoty PSM.
- Channel ID (2B) Udává identifikátor (CID) cílového kanálu, na který je paket doručen. V případě nespojované komunikace se jedná o konstantní hodnotu *0x0002*
- PSM ($\geq 2B$) „Protocol/Service Multiplexer“ Identifikuje použitý protokol vyšší vrstvy.
- Information payload obsahuje samotná data obdržená z vyšších protokolů při odesílání nebo předávaná vyšším protokolům po přijetí paketu.



Obrázek 3.6: Struktura paketu v základním L2CAP nespojovaném módu [13].

Spojovaná komunikace v „Retransmission“, „Flow Control“ a „Streaming módu“ Pro podporu ostatních módů komunikace jsou definovány rozšiřující hlavičky. „I-frame“ jsou používané pro výměnu dat informací mezi zařízeními. „S-frame“ slouží k potvrzení příchodu „I-frame“ nebo pro vyžádání znovu-odeslání daného „I-frame“. Jejich struktura je znázorněna na obrázku 3.7



Obrázek 3.7: Struktura paketů pro „Flow control“ a „Retransmission“ módy.

Protokol RFCOMM

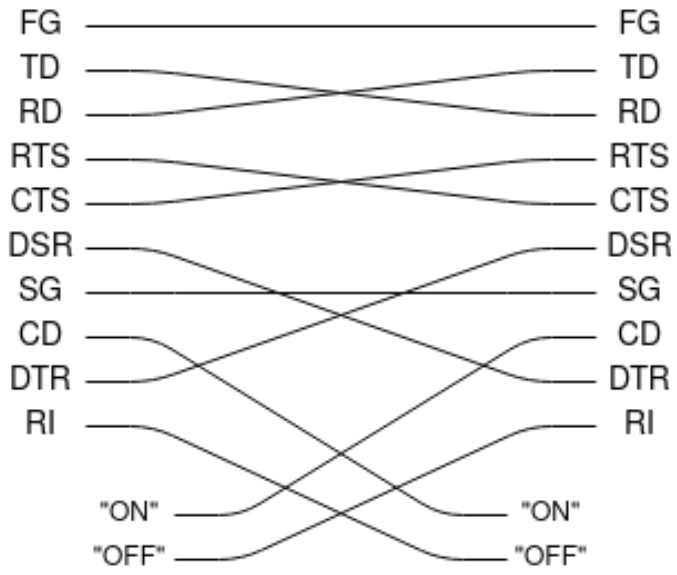
RFCOMM je protokol umožňující emulaci RS-232 sériového rozhraní pomocí L2CAP. Jedná se o jednoduchý protokol umožňující až 60 [10] současných spojení mezi dvěma Bluetooth zařízeními. Reálná hodnota současných spojení je dána implementací na konkrétním zařízení.

RS-232 Jedná se o sériové komunikační rozhraní obsahující 9 pinů. Jejich použití je následující [1]:

- *Data Carrier Detect (CD)* Informuje příjemce o skutečnosti, že jsou odesílány data.
- *Data Set Ready (DSR)* Je nastaven pokud je protistrana připravena přijímat data.
- *Receive Data Line (RD)* Pin pro přijímaní sériových dat.
- *Request to Send (RTS)* Signalizuje protistraně připravenost přijmout data.
- *Transmit Data Line (TD)* Linka pro odesílaní sériových dat.
- *Clear to Send (CTS)* Signalizace protistraně připravenost odeslat data.
- *Data Terminal Ready (DTR)* Označuje připravenost zařízení.
- *Ring Indicator (RI)* Indikuje příchozí zvonění (Používáno v modemech).
- *Signal Ground (SG)* Napětí na pinech je udáváno oproti referenční hodnotě na tomto pinu.
- *Frame Ground (FG)* Stínění

Propojení na obrázku 3.8 zobrazuje „null modem“[5] použitý ve specifikaci RFCOMM. Toto propojení by mělo fungovat ve většině případech užití tohoto protokolu.

Baud rate RFCOMM protokol neumožňuje spravování rychlosti v závislosti na nastavení „Baud rate“. Při nastavení specifické hodnoty se rychlost přenosu nemění [10]. Jedinou možností jak řídit rychlost přenosu je měnit rychlost odesílání dat v aplikaci, která tyto data odesílá.

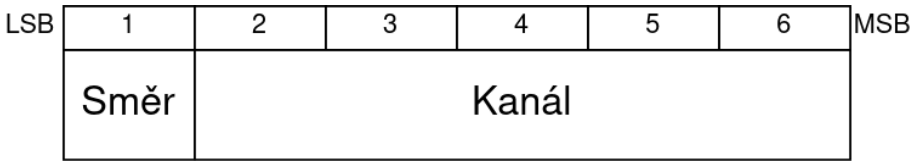


Obrázek 3.8: RFCOMM „null modem“ propojení [10].

[5] Propojení řídících signálu dvou RS-232 zařízení stejného typu

DLCI hodnota Bluetooth zařízením je umožněno otevřít až 60 [10] emulovaných sériových portů. Tato hodnota může být omezena specifickou implementací v použitém zařízení. Pro identifikaci jednotlivých spojení je určena hodnota DLCI „Data Link Connection Identifier“. Jedná se o šesti bitovou hodnotu jenž má použitelné hodnoty v rozmezí 2-61. Ostatní hodnoty jsou rezervované. Tato hodnota je unikátní v rámci jedné RFCOMM relace.

RFCOMM kanály Vzhledem ke skutečnosti, že aplikace na obou stranách komunikace mohou tvořit spojení nezávisle na sobě jsou hodnoty DLCI rozděleny mezi obě komunikující zařízení pomocí konceptu „RFCOMM server channels“. Tato šestibitová hodnota je rozdělena na jeden bit určující směr komunikace a 5 bitů určujících komunikační kanál. Struktura tohoto rozdělení je znázorněna na obrázku 3.9. Serverová aplikace používající RFCOMM protokol použije kanál jehož hodnota je v rozmezí 1-30. Tento kanál je následně zaregistrován v „Service Discovery Database“ pomocí Service Discovery Protocol. Ten je popsaný v kapitole 3.4. Zařízení zahajující RFCOMM relaci má bit určující směr komunikace v hodnotě 1. Druhé zařízení pak s hodnotou 0 [10].



Obrázek 3.9: Struktura hodnoty DLCI.

Zahájení spojení Při zahajování spojení je tedy vyčtena hodnota kanálu z SDP záznamu pro danou službu, se kterou se bude komunikovat. Následně je vytvořena hodnota DLCI ze získaného kanálu připojením směrového bitu (Získaného při zahájení RFCOMM relace). Takto vytvořená hodnota je následně odesílána s každým odchozím paketem.[10]

Protokol SDP

Service Discovery protocol je protokol umožňující zařízením vyhledávat požadované služby poskytované jinými zařízeními. SDP definuje jakým způsobem může zařízení vyhledávat služby podle určitých atributů bez vědomí o dostupných službách. Tento protokol používá komunikaci typu „Request/Response“, kdy každá interakce spočívá v odeslání jednoho požadavku PDU (Protocol Data Unit) a jedné PDU odpovědi.

Každé PDU obsahuje hlavičku a specifické parametry. Hlavička se skládá ze 3 hodnot [14]:

- *PDU ID* identifikátor typu paketu (velikost 1B)
- *TransactionID* unikátní identifikátor požadavku. Slouží ke spojení požadavku s odpovědí. (velikost 2B)
- *ParameterLength* udává délku parametrů v bytech obsažených v PDU (velikost 2B)

Některé odpovědi mohou být větší než maximální velikost PDU. V tomto případě odpověď obsahuje parametr „continuation state“ a dotazující zařízení může novým požadavkem s tímto parametrem získat zbylá data[14].

SDP záznam Všechny informace o dané službě jsou uchovávány na SDP serveru jako jeden záznam, který obsahuje list atributů dané služby. Některé mohou být společné mezi službami, ale je možné si definovat vlastní. Samotný atribut je složen ze dvou hodnot [14]:

- *AttributeID* unikátní v rámci služby. Udává význam hodnoty atributu.
- Samotná hodnota atributu s proměnnou délkou. Obsahuje hlavičku s typem dat a jejich délkou.

Service Class Každá služba je instancí jedné třídy služeb. Ta udává definice jednotlivých atributů a specifikuje jejich použití. Každá třída služeb má unikátní identifikátor (UUID).

Knihovna bluez

Knihovna bluez je oficiální knihovna určená pro linuxové systémy implementující „Bluetooth protocol stack“. Obsahuje relativně nízkou míru abstrakce. Umožňuje pracovat se všemi protokoly Bluetooth protocol stacku. Samotná komunikace po připojení k zařízení probíhá za použití socketů. Tato komunikace se velice podobá síťové komunikaci. Od verze knihovny 5.0 jsou některé funkce knihovny označené za „Deprecated“ a není poskytnuta náhrada. Jejich funkcionalita byla přesunuta do DBUS api.

DBUS API

Knihovna bluez také poskytuje API pomocí systému zpráv DBUS, který je popsaný v kapitole 3.6. Toto rozhraní obsahuje vyšší míru abstrakce. Interakce probíhá pomocí volání zpřístupněných funkcí a registrací objektů, které implementují rozhraní poskytnutá touto knihovnou. Nad takto registrovanými objekty následně bluez volá funkce, které informují program o příchozích spojeních.

3.5 Protokol MIDI

Protokol MIDI (Musical Instrument Digital Interface) je standard umožňující reprezentaci informací o hudebním vystoupení jako data. Ta jsou přenášena pomocí MIDI zpráv. Zprávy udávají informace, pomocí kterých může být hudební vystoupení replikováno s použitím syntetizéru. Ten přijímá MIDI zprávy a generuje samotný zvuk. Každé propojené zařízení může přijímat zprávy na jednom z 16 logických kanálů [2].

MIDI zprávy MIDI zprávy jsou složeny z 8bitového statusu, za kterým většinou následují dva datové byty. Struktura midi zpráv je zobrazena na obrázku 3.10. Na nejvyšší úrovni jsou zprávy děleny na Kanálové a Systémové.

Kanálové zprávy, oproti systémovým, přísluší k určitému kanálu a tato informace je obsažena ve stavovém bytu. Tyto zprávy jsou následně děleny na obsahující informace o zvuku (Channel Voice) a modifikující jak příjemce reaguje na zvukové zprávy (Channel Mode).

Zvukové zprávy (Channel Voice)

Obsahují informace o zvukovém dění. Mezi ně patří:

- Note On

Obrázek 3.10: Struktura midi zpráv [2].

- Note Off
- Polyphonic Key Pressure
- Channel Pressure
- Pitch Bend Change
- Program Change
- Control Change

Status byte jednotlivých zpráv je v tabulce 3.1.

Typ zprávy	Status byte
Note-Off	0x8n
Note-On	0x9n
Poly Key Pressure	0xAn
Control Change	0xBn
Program Change	0xCn
Channel Pressure	0xDn
Pitch Bend	0xEn

Tabulka 3.1: Status byte „Channel Voice“ zpráv [2].

Note On / Note Off zprávy Tyto typ zpráv označuje začátek a konec zvuku daného tónu. Jsou odeslány při stisku nebo uvolnění klávesy. Při stisku se odesílá zpráva typu Note On. Ta může být odeslána na jednom z 16 kanálu a identifikátor tohoto kanálu je odeslán ve status bytu. Po status bytu následuje byte identifikující stisknutou klávesu a další byte specifikující rychlost s jakou byla klávesa stisknuta. Identifikátor klávesy udává tón a rychlost stisku většinou udává amplitudu. Při uvolnění klávesy se odesílá zpráva Note Off, která má stejnou strukturu. Identifikátor klávesy udává, která klávesa byla uvolněna. Byte rychlosti udává rychlost s jakou byla uvolněna. Tato informace je většinou ignorována [2].

Zprávy Polyphonic Key Pressure / Channel Pressure Některé hudební nástroje obsahují tlakový senzor, který umožňuje po stisknutí klávesy dále upravovat produkovaný

tón tlakem. Pokud hudební nástroj obsahuje tlakový senzor pro každou klávesu, informace o tlaku, se kterým je klávesa stlačena, je odeslán pomocí zprávy „Polyphonic Key Pressure“. Ta obsahuje identifikátor klávesy a tlak, se kterým je stlačena [2]. Pokud hudební nástroj obsahuje pouze jeden tlakový senzor pro všechny klávesy. Informace o tlaku je posílána pomocí „Channel Pressure“. Tato zpráva obsahuje pouze byte s informací o tlaku.

Zprávy Pitch Bend Tento typ zpráv je odesílán pokud nastane změněna pozice ovladače tónu „pitch bend wheel“. Slouží k posunu tónu, který je přehráván na daném kanálu. Hodnota posunutí je udávána pomocí dvou bytů [2] pro umožnění jemnějšího rozlišení.

Zpráva Program Change Zpráva „Program Change“ se používá pro změnu nástroje na určitém kanálu. Obsahuje status byte, ve kterém je ID kanálu a byte specifikující nové číslo programu [2].

Zpráva Control Change Tyto zprávy jsou určeny k ovládání různých hodnot v syntetizéru a měly by, jako ostatní Channel zprávy, ovlivňovat pouze kanál, na kterém jsou posílány. Zpráva obsahuje dva byty. Jeden pro identifikaci hodnoty, kterou je potřeba změnit a druhý pro novou hodnotu [2].

Zprávy Channel Mode

Typ zprávy	Status byte
All Sound Off	120
Reset All Controllers	121
Local Control	122
All Notest Off	123
Omni Off	124
Omni On	125
Mono On(Poly Off)	126
Poly On(Mono Off)	127

Tabulka 3.2: Status byte „Channel Mode“ zpráv [2].

Tyto zprávy slouží k modifikaci jak syntetizér reaguje na přijaté MIDI zprávy. Hodnota 121 resetuje všechny midi ovladače. Hodnota 122 umožňuje zapnutí/vypnutí ovladače u syntetizérů s vlastním ovladačem. Hodnoty 124-127 jsou použity pro zapnutí nebo vypnutí „Omni módu“ a přepnutí mezi „Mono“ a „Poly“ módem [2].

Pokud je zapnut „Omni mód“, syntetizér odpovídá na zprávy na jakémkoliv kanálu. Pokud je vypnut, syntetizér odpovídá na zprávy pouze na jednom kanálu. Poly mód umožňuje hrát několik tónů zároveň. Mono mód umožňuje hru pouze jednoho tónu ve stejný čas. Status byte těchto zpráv jsou v tabulce 3.2.

Systémové zprávy

System Common zprávy Tyto zprávy jsou určeny pro všechny příjemce v propojeném systému. „MTC Quarter Frame“ slouží k synchronizaci všech propojených MIDI zařízení. „Song Select“ slouží pro MIDI zařízení jako je třeba „Drum machine“, které mají v paměti několik skladeb. Zpráva „Song Position Pointer“ poté umožňuje nastavit pozici v dané

skladbě. Hodnota této zprávy udává pozici, která je dána hodinovými cykly MIDI od začátku skladby. „Tune Request“ slouží pro analogové syntetizéry jako požadavek o naladění interních oscilátorů. Zpráva „EOX“ udává konec System Exclusive zprávy, která má variabilní velikost [2]. Status byte těchto zpráv je v tabulce 3.3.

Typ zprávy	Status byte
MIDI Time Code Quarter Frame	0xF1
Song Position Pointer	0xF2
Song Select	0xF3
Tune Request	0xF6
EOX (End of Exclusive)	0xF7

Tabulka 3.3: Status byte „System Common“ zpráv [2].

System Real Time zprávy Tyto zprávy slouží k synchronizaci všech časově závislých zařízení v MIDI systému a mají přednost před všemi ostatními typy. Můžou se objevit i uprostřed odesílaní zprávy jiného typu [2]. Typ „Timing Clock“ nastavuje tempo. „Start“, „Continue“, „Stop“ slouží k ovládání přehrávání skladby. Status byte těchto zpráv je v tabulce 3.4.

Typ zprávy	Status byte
Timing Clock	0xF8
Start	0xFA
Continue	0xFB
Stop	0xFC
Active Sensing	0xFE
System Reset	0xFF

Tabulka 3.4: Status byte „System Real Time“ zpráv [2].

„Active Sensing signal“ slouží k eliminaci spuštěných not, po odpojení kabelu. Kdy před odpojením byla odeslána zpráva Note On a tím zapnuto přehrávání daného tónu. Po odpojení už nepřijde zpráva Note Off a tón by byl zapnut do nekonečna.

„System Reset“ slouží k restartu a inicializaci všech zařízení, která přijmou tuto zprávu [2]. Většinou posílána uživatelem.

System Exclusive Zprávy

Tyto zprávy jsou používány pro specifickou komunikaci mezi jednotlivými zařízeními. Jejich formát je závislý na výrobci, ale většinou obsahují ID, které tomuto výrobci bylo přiděleno. Po tomto ID následuje libovolné množství datových bytů. Zpráva je ukončena odesláním EOX [2]. Status byte této zprávy je v tabulce 3.5.

Typ zprávy	Status byte
System Exclusive	0xF0

Tabulka 3.5: Status byte „System Exclusive“ zpráv[2].

3.6 Systém D-Bus

D-Bus je systém zprostředkovávající mezi procesovou komunikaci se snahou o jednoduchost použití. Pro komunikaci je použit binární protokol a je tedy méně náročný oproti obdobným technologiím používajícím textový protokol, z důvodu absence zdlouhavé serializace.

Základní specifikace D-Bus protokolu specifikuje pouze připojení dvou zařízení jako peer-to-peer [11]. Jedna aplikace může tedy komunikovat pouze s jedním dalším příjemcem. Komunikace mezi více příjemci probíhá skrze hlavní aplikaci „message bus“. Ta přijímá připojení od ostatních a přeposílá zprávy mezi nimi.

D-Bus má dvě specifické použití. „System bus“ pro notifikace od systému pro uživatele nebo pro umožnění systému vyžádáni vstupu od uživatele. „Session bus“ slouží pro použití v rámci desktopových prostředí jako například GNOME nebo KDE.

Typový systém

D-bus používá typový systém, který umožňuje jakékoliv datové typy reprezentovat jako sekvenci bytů. Takto vytvořená sekvence již neobsahuje informace o datovém typu. Obsahuje pouze informace o struktuře, která je zapsána pomocí ASCII znaků [11]. Ta tvoří samotný datový typ. Tuto strukturu může tvořit základní datový typ, jejich pole nebo struktura obsahující další datové typy.

Základní datové typy Základní datové typy jsou definovány jako jeden ASCII znak. Jsou rozděleny na typy s fixní velikostí a řetězcové. Řetězcové jsou kódované pomocí UTF-8. Řetězec je ukončen pomocí jednoho NULL bytu [11]. Tento byte není součástí odesílaného textu.

Název	ASCII znak	Kódování
Byte	y	Bezznaménkový 8-bitový integer
Boolean	b	Boolovská hodnota: 0=false, 1=true
Int16	n	16-bitový integer ve dvojkovém doplňkovém kódu
Uint16	q	Bezznaménkový 16-bitový integer
Int32	i	32-bitový integer ve dvojkovém doplňkovém kódu
Uint32	u	Bezznaménkový 32-bitový integer
Int64	x	64-bitový integer ve dvojkovém doplňku
Uint64	t	Bezznaménkový 64-bitový integer
Double	d	IEEE 754 hodnota s plovoucí desetinou čárkou
Unix_FD	h	Bezznaménkový 32-bitový integer reprezentující index do „file descriptor“ pole

Tabulka 3.6: Seznam datových typů s fixní délkou [11].

Název	ASCII znak	Kódování
String	s	
Object_path	o	Syntakticky správná cesta k objektu 3.6
Signature	g	Žádný nebo více specifikací datového typu

Tabulka 3.7: Seznam řetězcových datových typů [11].

Cesta k objektu Cesta k objektu slouží jako odkaz na jeho instanci. Veškeré objekty tvoří hierarchický strom. Cesta většinou začíná pomocí reverzního doménového jména a obsahuje název implementovaného rozhraní a verzi. Touto syntaxí je umožněno aplikacím provozovat více verzí služby ve stejném procesu.

Cesta muže mít libovolnou velikost, ale musí začínat znakem „/“ a mít bloky oddělené tímto znakem. Každý blok může obsahovat pouze znaky A-Z a-z 0-9. Žádný blok nesmí být prázdný a cesta nemůže končit pomocí „/“, pokud se nejedná o kořenovou cestu [11].

Kontejnerové typy Tyto typy seskupují více datových typů do logických celků. Jsou definovány 4 kontejnerové typy [11]:

- STRUCT
- ARRAY
- VARIANT
- DICT_ENTRY

STRUCT Typ struktury má ASCII kód „r“, ale ten se neobjevuje v signatuře datového typu. Používají se znaky kulatých závorek „(“, „)“ pro označení začátku a konce.

Takto je zapsána struktura obsahující dvě 32-bitové znaménkové čísla: *(ii)*. Struktury mohou být i zanořovány: *(i(ii))*.

ARRAY Pole je označeno ASCII znakem „a“. Tento znak musí být následován identifikátorem typu, který udává typ pro všechny elementy pole. Tento typ může být i další definice kontejnerového typu. Definice pole může vypadat následovně:

- Pole obsahující 32-bitové znaménkové číslo: *ai*.
- Pole obsahující strukturu o dvou 32-bitových znaménkových číslech: *a(ii)*.

VARIANT Označený pomocí znaku „v“. Jedná se o libovolná datový typ a obsahuje definici a hodnotu. Při použití datových typů variant nesmí hloubka zanoření přesáhnout hodnotu 64 [11].

DICT_ENTRY Datový typ tvořící slovník. Podobá se typu struktury. Pro označení začátku a konce se používají složené závorky „{“, „}“. Zavádí omezující podmínky [11]:

- Musí být použit pouze jako element v poli.
- Obsahuje pouze dva datové typy udávající klíč a hodnotu.
- Klíč musí být základní datový typ. Nemůže se jednat o kontejnerový typ.

3.7 Architektura MVVM

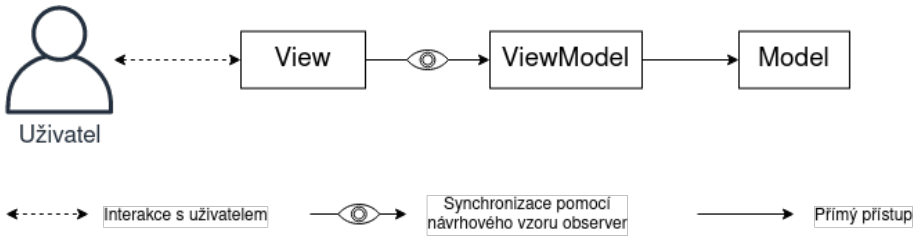
Model-View-ViewModel je návrhový vzor, který umožňuje rozdělení aplikace na logické celky. Každý celek zprostředkovává jednu funkcionalitu aplikace.

View Tato část aplikace je zodpovědná za uživatelské prostředí a zobrazování dat. Je to jediná část, která provádí interakci s uživatelem [18]. Definuje se zde vzhled aplikace. Data ve View jsou pomocí technologie data binding zobrazována z ViewModelu. Interakce uživatele je propagována do ViewModelu.

ViewModel ViewModel se stará o zobrazovaná data a stavy aplikace. View je notifikován o všech změnách dat ve ViewModelu a provádí obnovení zobrazených dat. ViewModel obsahuje referenci na model aplikace a může tedy provádět volání servisních metod nebo přistupovat k databázi [18].

Model Model zahrnuje všechna dostupná data (Například databáze). Při změně dat je notifikován ViewModel, který si tyto změny vyžádá.

Shrnutí Klíčová funkcionalita MVVM návrhového vzoru je použití data-binding technologie za pomocí návrhového vzoru Observer [18]. Tímto je umožněno synchronizovat zobrazovaná data s aktuálními hodnotami, která jsou aplikaci k dispozici.



Obrázek 3.11: Návrhový vzor MVVM [18].

3.8 Android LiveData

Android LiveData je technologie implementující návrhový vzor observer. Tato technologie je obeznámena s životním cyklem ostatních komponent aplikace. Tím je zajištěna notifikace pouze těch „Observerů“, které jsou ve správném stavu životního cyklu [8]. Pokud je ukončen životní cyklus LiveData objektu, je uvolněna paměť, která je s tímto objektem spojena. Tím se předchází úniku paměti.

Pokud je aktivita (Obrazovka aplikace) registrovaná k přijímání změn dat a je neaktivní, nedostává notifikace o změně dat. Pokud je znovu aktivní, data zobrazena v této aktivitě jsou obnovena na aktuální hodnoty.

Transformace LiveData

Technologie LiveData umožňuje provádět transformace nad uloženými hodnotami. Tyto transformace jsou vypočítávány pouze pokud se změní hodnota a jsou registrovány objekty, které mají být o změně informovány [8]. Tím zamezuje provádění výpočtů, kdy výsledek není použit.

Kapitola 4

Výběr použitých technologií a návrh řešení

Na základě předchozích kapitol, ve kterých byly shrnuty existující ovladače a současné technologie, jsou v aktuální kapitole tyto technologie porovnány. Dále jsou zde specifikovány vlastnosti, které bude výsledný pedálový ovladač splňovat.

4.1 Porovnání ovladačů existujících hudebních komb

V kapitole 2 je shrnuto několik existujících ovladačů a kytarových komb. Jejich vlastnosti byly zaneseny pro přehlednost do tabulky 4.1.

	Yamaha THR10 II	Boss Katana-Air	Marshall DSL40CR	Line 6 HX stomp
Mobilní aplikace	Ano	Ano	Ne	Ne
Pedálový ovladač	Ne	Ne	Ano	Ano
PC aplikace	Ano	Ne	Ne	Ano
Uložení nastavení	Ano	Ano	Ne	Ano
Open Source	Ne	Ne	Ne	Ne
Bezdrátové ovládání	Ano	Ano	Ne	Ne
Cena	$412	$409	$750	$600

Tabulka 4.1: Porovnání funkcí jednotlivých komb

Yamaga THR10 II Na základě hodnot z tabulky 4.1 je možné vidět, že kombo Yamaha THR10 II splňuje nejvíce mnou porovnávaných vlastností. Toto kombo však postrádá pedálový ovladač a tím pádem neumožňuje jednoduchou změnu profilu bez použití mobilní aplikace. Tu je možné připojit bezdrátově pomocí Bluetooth a uživateli je umožněno konfigurovat a ukládat nastavení do jedné z pěti pozic. Kombo neumožňuje uživateli uložit více jak pět nastavení. Po zaplnění těchto pozic musí uživatel některé nastavení přepsat.

Boss Katana-Air Kombo Boss Katana-Air umožňuje připojení hudebního nástroje pomocí bezdrátové technologie. Mobilní aplikace pro konfiguraci neposkytuje tak detailní konfiguraci tónu jako aplikace pro Yamaha THR10 II, ale umožňuje ukládat až 30 přednastavených konfigurací. Dále také obsahuje některé funkce navíc. Jako je třeba ukládání nastavení

do cloudu nebo integrovanou ladičku. Toto kombo také nedisponuje pedálovým ovladačem a nelze jej ovládat přes počítač kvůli absenci konfigurační aplikace.

Marshall DSL40CR Kytarové kombo Marshall DSL40CR je lampové a je tedy podstatně dražší. Nedisponuje však možností konfigurace pomocí mobilního telefonu nebo PC. Je možné ho ovládat pomocí knoflíků přímo na zařízení, které má dva kanály. Mezi nimi je možné přepínat pomocí přiloženého pedálového přepínače. Kombo s ovladačem komunikuje pomocí midi a základní pedálový přepínač je možné zaměnit za rozšířenější, který umožňuje přepínání více módů.

4.2 Zaměření práce

Práci jsem si vybral, jelikož kytarové kombo, které vlastním, neumožňuje připojení pedálového ovladače. Je tedy velmi složité v průběhu hudební skladby měnit kytarový tón. Nová kytarová komba s touto funkcionalitou mají relativně vysokou cenu. Dále jsem tuto příležitost využil pro poznání technologií komunikace hudebních nástrojů a vývoje mobilních aplikací.

4.3 Technická specifikace výrobku

Je nutné vytvořit pedálový ovladač pro ovládání kytarového komba. Ovladač bude obsahovat dvě tlačítka pro výběr přednastavené konfigurace. Také je nutné vytvořit mobilní aplikaci sloužící pro vytvoření těchto konfigurací. Výsledná práce musí splňovat tyto technické parametry:

Open source Práce by měla mít otevřený zdrojový kód. To umožní uživatelům stáhnout si zdrojový kód a případně upravit ke svému specifickému použití. Tím je možné místo koupení zesilovače, který podporuje pedálový přepínač, rozšířit funkcionalitu stávajícího zesilovače.

Jednoduché na ovládání Aby bylo možné používat ovladač i při hře na hudební nástroj je nutné, aby bylo ovládání jednoduché. Měla by být minimalizována možnost nechtěného provedení změny při jeho použití.

Umožnit komplexní nastavení Pokud by ovladač neumožňoval nastavení všech parametrů zesilovače, značně by tím byla limitována jeho možnost využití. Toto nastavení by mělo být proveditelné před samotným začátkem hudebního vystoupení, aby byla dodržena vlastnost jednoduchého ovládání.

Nastavení pomocí mobilní aplikace Aby byl ovladač co nejjednodušší a tím se zamezila možnost nechtěného přenastavení, konfiguraci by mělo být možno provést pomocí oddělené aplikace. Nejlépe pomocí mobilního telefonu, který je přenosný a umožňuje přívětivou interakci s uživatelem.

Funkčnost i bez mobilní aplikace Samotný ovladač by měl být schopen pracovat i bez spojení s mobilní aplikací. Ta by měla sloužit pouze k nastavení. Po dokončení by nemělo odpojení ovlivnit funkcionalitu pedálového ovladače.

Přívětivé uživatelské prostředí mobilní aplikace Uživatelské prostředí mobilní aplikace musí být přehledné a snadno použitelné. Také by mělo být rozčleněné do logických celků, aby uživatel věděl kde hledat určité nastavení.

Implementační jazyk Pro implementaci softwaru bude použit programovací jazyk Kotlin, který umožňuje kompilaci do byte-kódu spustitelného na virtuálních strojích jazyku Java a také do nativního kódu dané architektury procesoru. Nativní kompilace bude použita pro pedálový ovladač, díky nižšímu času startu aplikace. Kompilace do byte-kódu bude použita pro mobilní aplikaci.

Rozměry Hardwarová část ovladače se musí vměstnat do krabičky typu *1590BB* o rozměrech *119mm x 94mm x 34mm*. Použijte vhodný hardware pro implementaci.

Ukládání konfigurace Uživatelská konfigurace bude uložena v paměti ovladače ve vhodném formátu a po vybrání bude odeslána do kytarového komba *YAMAHA THR10* pomocí rozhraní USB s předem neznámým protokolem. Tento protokol je nutné replikovat analýzou komunikace komba a jeho konfigurační aplikace.

Konfigurace Kombo se po přijetí zprávy nastaví na danou konfiguraci, jejíž tvorba bude provedena na mobilním zařízení pomocí vytvořené aplikace. Ta bude s pedálovým ovladačem komunikovat pomocí vhodně zvolené technologie a komunikačního protokolu.

Mobilní aplikace Aplikace bude mít uživatelsky přívětivé rozhraní. Kromě tvorby konfigurace bude umožňovat přímé ovládání komba, kdy budou nastavení zobrazená v aplikaci synchronizována s aktuálními hodnotami komba.

Bezdrátové propojení s mobilní aplikací Samotné spojení s mobilní aplikací by mělo být bezdrátové, jelikož je pohodlnější pro uživatele. Není nutné aby uživatel vlastnil propojující kabel.

Napájení Ovladač bude napájen pomocí bateriového zdroje. Je nutné, aby byl schopen na tomto napájení pracovat alespoň několik hodin, aby bylo možné jej použít při hudebním vystoupení. Napájení pedálového ovladače a informace o stisku tlačítek bude spravováno dodanou rozšiřující deskou. Informace o napájení jako je stav baterie, doba běhu, proudový odběr nebo stav nabíjení je možné zjistit komunikací s deskou. Ta probíhá skrze rozhraní *UART* za použití dodaného protokolu.

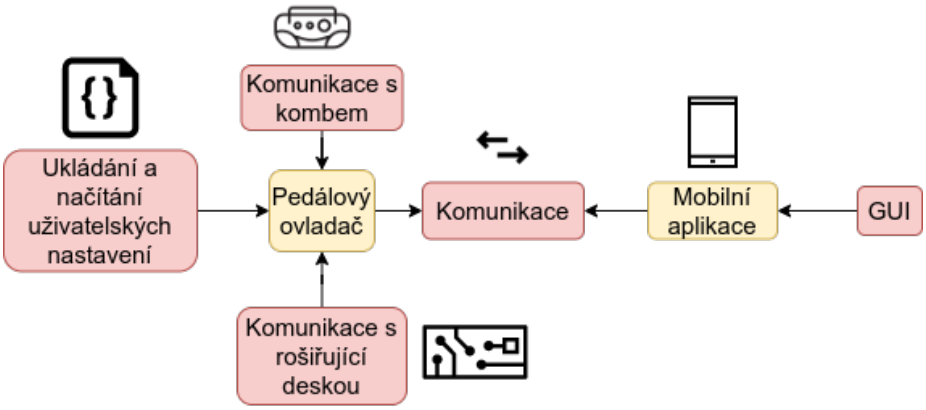
Kapitola 5

Realizace ovladače a mobilní aplikace

V této kapitole je popsáno rozdělení celkového problému na menší spolupracující logické bloky. Je zde popsán postup, který byl uplatněn při implementaci jednotlivých logických bloků a řešení jednotlivých problémů spojených s realizací celkového systému. Na konci této kapitoly jsou popsány testy splnění vlastností systému specifikovaných v kapitole 4.3 a jejich výsledky.

5.1 Koncepce systému

Systém pedálového ovladače a mobilní aplikace byl rozdělen do několika na sobě závislých bloků znázorněných v diagramu 5.1.



Obrázek 5.1: Diagram struktury práce.

Komunikace s kombem Komunikace s kytarovým kombem je prováděna pomocí neznámého protokolu po rozhraní USB. Je nutné tomuto protokolu porozumět a komunikaci replikovat pro použití v pedálovém ovladači.

Komunikace s rozšiřující deskou Tato komunikace probíhá po rozhraní UART se známým protokolem. Je nutné komunikaci po tomto rozhraní implementovat.

Ukládání a načítání uživatelských nastavení Jelikož pedálový ovladač musí pracovat nezávisle na připojení mobilní aplikace musí být uživatelské nastavení uloženy v pedálovém ovladači. Aby takové uložení bylo možné, je nutné data převést do vhodného formátu a umožnit programu zpětné načtení těchto dat.

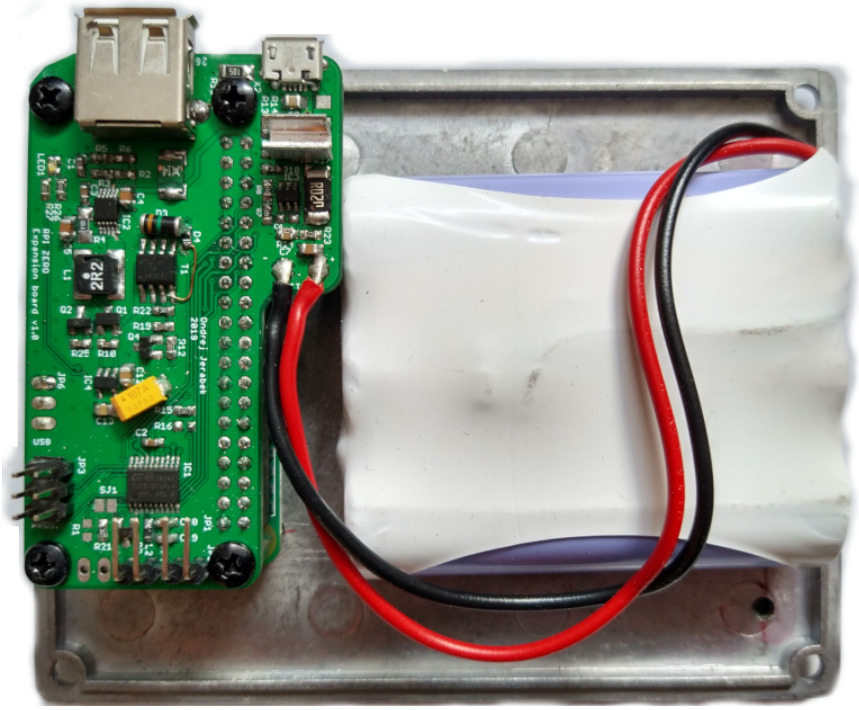
Komunikace s mobilní aplikací Pro tuto komunikaci je nutné vybrat vhodnou technologii a vytvořit komunikační protokol, kterému budou obě strany rozumět. Dále je potřeba vytvořit obsluhu požadavků přijímaných z mobilní aplikace.

5.2 Tvorba prototypu pedálového ovladače

V této kapitole je shrnuta tvorba pedálového ovladače.

Hardware ovladače

Jako hardware ovladače byl použit jednodeskový mikropočítač *Raspberry Pi Zero W*, jelikož poskytuje relativně vysokou abstrakci a umožňuje použití již existujících knihoven pro operační systém linux. Při použití vývojové platformy *Arduino* by bylo nutné propojit veškeré periferie zvlášť a poté řešit komunikaci s nimi. Jelikož Arduino nepodporuje v základním provedení bezdrátovou komunikaci, bylo by potřeba tuto funkcionalitu doplnit rozšiřující deskou. Mikropočítač Raspberry Pi Zero W má několikanásobně vyšší výkon, jeho nevýhodou je však relativně dlouhá doba pro nastartování operačního systému.



Obrázek 5.2: Hardware pedálového ovladače.

Napájení hardwaru ovladače

Samotný mikropočítač je nutné napájet. Jelikož je napájen pomocí micro-USB konektoru je mikropočítač napojen na rozšiřující desku. Ta zprostředkovává zapnutí/vypnutí napájení a nabíjení baterie.

Ovládání hardwaru

Jelikož zapínaní a vypínání je řešeno stiskem tlačítka pro přepínání uživatelské konfigurace po určitou dobu, jsou tyto tlačítka připojena k mikropočítači skrze rozšiřující desku. Ta při detekci stisknutí nebo uvolnění tlačítka odesílá tuto informaci do mikropočítače. Při požadavku na vypnutí ovladače (Podržení pravého tlačítka změny konfigurace) je odeslán mikropočítači signál a software se připraví a následně vypne operační systém. Zapnutí ovladače z důvodu startování operačního systému trvá delší dobu. Doba náběhu operačního systému a jeho optimalizace je popsána v kapitole 5.4

5.3 Komunikace s rozšiřující deskou

Jelikož je rozšiřující deska pevně spjata s mikropočítačem je komunikační linka s rozšiřující deskou inicializována hned po startu aplikace. Ta probíhá v samostatném vlákně, jelikož je potřeba čekat na příchozí zprávy. Při příchodu zprávy je převedena do podoby objektu a následně je předána hlavnímu vláknu aplikace, které provádí obsluhu.

Komunikační protokol

Komunikační protokol je textový. Znaky jsou v ASCII kódování. Každá zpráva začíná pomocí znaku „$“. Následuje název příkazu a jeho parametry oddělené znakem „;“ Protokol se skládá ze dvou typů zpráv.

- *Informační* sloužící k odesílání informací z rozšiřující desky. U tohoto typu zpráv není očekávána odpověď. Mezi ně patří zpráva o vypnutí a změna stavu tlačítka.
- Ostatní zprávy patří to kategorie požadavků. Ty je nutné odeslat a následně čekat na odpověď, ve které jsou obsaženy požadované informace.

Zpráva vypnutí Zpráva `$off` oznamuje aplikaci příkaz pro ukončení činnosti a vypnutí systému. Je odeslána pokud uživatel podrží pravé tlačítko přepnutí konfigurace a tím vypne ovladač.

Změna stavu tlačítka Zpráva `$btn` informuje aplikaci o stisku tlačítka. Má dva parametry, ve kterých je uvedeno ID tlačítka, které bylo stisknuto a zda bylo stisknuto nebo uvolněno. Pokud se jedná o zprávu o uvolnění tlačítka, obsahuje i třetí parametr, který udává jak dlouho bylo tlačítko stisknuto.

Stav systému Pro získání stavu je nutné odeslat požadavek `$sta`. Odpověď má následující strukturu: `$ok;1254;74;dis;310`.

První parametr odpovědi je čas běhu (Uptime) v sekundách. Druhý parametr je kapacita baterie v procentech. Třetí udává stav nabíjení. Může nabývat hodnot „dis“,„chg“,„end“. Poslední parametr udává aktuální proud, který mikropočítač odebírá.

Verze firmware Pro získání verze firmware je možné odeslat zprávu ve formátu `$fwv`. Odpověď následně obsahuje tři parametry udávající hlavní, vedlejší verzi a verzi záplaty. Může vypadat následovně. `$ok;1;1;0`

Watchdog zpráva Při běhu aplikace je potřeba každých alespoň 5 sekund odesílat zprávu o jejím běhu, jinak je mikropočítač restartován. Tato zpráva má následující formát: `$hbt`. Po přijmutí je odesílána odpověď `$ok`.

Vypnutí napájení Pokud aplikace požaduje vypnutí napájení, odesílá se požadavek `$off`. Odpověď na tento požadavek je `$ok $off`. Při příchodu této odpovědi by se měla aplikace ukončit a vypnout systém. Jelikož rozšiřující deska vypne mikropočítač do 15 sekund od napájení nebo ihned po tom co detekuje vypínací sekvenci mikropočítače.

5.4 Spuštění systému ovladače

Na mikropočítači raspberry pi zero w, který se nachází v ovladači je nainstalován operační systém raspbian. Tento systém obsahuje relativně vysokou abstrakci a tak usnadňuje práci s hardwarovými zařízeními. Také umožňuje použití systémových knihoven a spoustu již vytvořených programů.

Použitý mikropočítač má relativně nízký výkon a start tohoto systému může trvat v rámci desítek sekund. Aktuální doba startu softwaru ovladače je přibližně 32 sekund. Ta je příliš dlouhá a uživatel při každém startu musí čekat. Aby bylo použití ovladače pohodlnější je nutné tento start optimalizovat.

SD-karta

Tato optimalizace byla provedena použitím SD-karty (na které se nachází systém) s vyšším rychlostním označením. Nyní je systém uložen na SD-kartě s označením *class 10*. Tato změna způsobila zrychlení náběhu o několik sekund.

Systémové služby

Další optimalizace byla provedena zakázáním spuštění nepotřebných systémových služeb po startu. Pomocí příkazu `systemd-analyze` je možné zjistit dobu startu systému. Aktuální výstup je následující.

```
Startup finished in 4.502s (kernel) + 32.285s (userspace) = 36.788s
graphical.target reached after 32.075s in userspace
```

Pomocí příkazu `systemd-analyze blame` je možné zjistit dopad jednotlivých služeb na dobu startu systému. Aktuální stav je následující:

```
9.506s dhcpcd.service
9.122s dev-mmcblk0p2.device
5.647s hciuart.service
3.508s keyboard-setup.service
3.447s systemd-udev-trigger.service
2.759s raspi-config.service
2.573s dphys-swapfile.service
```

```
2.438s networking.service
2.066s systemd-journald.service
2.055s rpi-eeprom-update.service
2.016s wpa_supplicant.service
1.655s systemd-fsck@dev-disk-by\x2dpartuuid-58625724\x2d01.service
1.610s rng-tools.service
1.487s systemd-timesyncd.service
...
```

Nyní je možné jednotlivé služby projít a zrušit automatické spouštění těch co nejsou potřeba pro běh ovladače.

dhcpcd.service Tato služba zprostředkovává DHCP a ovladač ji nepotřebuje, jelikož ke komunikaci se používá Bluetooth. Přístup do konzole zařízení je pomoci síťového připojení a zakázání této služby nyní by způsobilo zamezení přístupu do zařízení, protože by nemělo přidělenou IP adresu. Prozatím je možné pomocí konfiguračního nástroje `raspi-config` nastavit, aby DHCP neblokovalo postup náběhu systému.

dev-mmcblk0p2.device Tato položka udává dobu připojení oddílu SD karty. Momentálně není možné toto připojení optimalizovat. Jednou z možností by bylo pořízení rychlejší SD karty.

hciuart.service Služba hciuart obstarává konfiguraci modemu Bluetooth připojeného pomocí rozhraní UART. Ovladač Bluetooth používá k připojení k mobilní aplikaci a není tedy možné tuto službu zakázat.

keyboard-setup.service Jelikož k pedálovému ovladači nebude připojena klávesnice, není tato služba, která slouží k nastavení správného rozložení klávesnice, potřeba.

systemd-udev-trigger.service Služba sloužící k detekci a obsluze připojení USB zařízení. Kombo je připojeno pomocí USB, ale detekce připojení je kontrolována manuálně. Tato služba tedy není potřeba.

raspi-config.service Raspi-config je program, který nastavuje raspberry pi po startu. Tato služba tedy nebude zakázána.

dphys-swapfile.service Služba obstarávající swap. Na zařízení poběží jediná aplikace a spousta služeb bude zakázána. Swap tedy není potřeba a je možné tuto službu zakázat.

networking.service Jelikož ovladač nepoužívá síťové spojení je možné tuto službu vypnout. Pro účely konfigurace zařízení, která probíhá po síti, je tato služba prozatím ponechána zapnuta.

Vypnutí grafického prostředí a přetaktování Po optimalizaci systémových služeb byla vypnuta podpora grafického prostředí a zařízení bylo přetaktováno.

Výsledek Optimalizacemi bylo dosaženo následujícího času náběhu systému.

```
Startup finished in 3.564s (kernel) + 19.115s (userspace) = 22.680s
multi-user.target reached after 18.092s in userspace
```

Jak je možné vidět, čas náběhu systému byl zkrácen o přibližně 14 sekund. Po dokončení konfigurace zařízení je možné zakázat další služby obstarávající připojení k síti. Tato optimalizace zkrátí tuto dobu o další přibližně 4 sekundy. Pro další zrychlení by bylo nutné pořídit rychlejší SD-kartu.

5.5 Popis komunikačního protokolu hudebního komba

Pro komunikaci s kytarovým kombem je nutné znát protokol, pomocí kterého probíhá komunikace s konfigurační aplikací. Při připojení kytarového komba pomocí USB jsou vytvořeny audio vstupy a výstupy, ale také MIDI zařízení. Při sledování komunikace na midi rozhraní je možné pozorovat příchod dat při pootočením knoflíků nastavení komba.

Komunikace s kombem

Zde je příklad komunikace přijímané z komba.

```
0xF0 0x43 0x7D 0x10 0x41 0x30 0x01 0x01 0x00 0x35 0xF7
0xF0 0x43 0x7D 0x10 0x41 0x30 0x01 0x02 0x00 0x64 0xF7
0xF0 0x43 0x7D 0x10 0x41 0x30 0x01 0x03 0x00 0x44 0xF7
0xF0 0x43 0x7D 0x10 0x41 0x30 0x01 0x04 0x00 0x61 0xF7
0xF0 0x43 0x7D 0x10 0x41 0x30 0x01 0x05 0x00 0x4B 0xF7
0xF0 0x43 0x7D 0x60 0x44 0x54 0x41 0x31 0xF7
0xF0 0x43 0x7D 0x60 0x44 0x54 0x41 0x31 0xF7
0xF0 0x43 0x7D 0x60 0x44 0x54 0x41 0x31 0xF7
```

Midi protokol Jak je možné vidět z příkladu komunikace, jedná se o data odpovídající protokolu MIDI za pomocí system exclusive zpráv. Tyto zprávy jsou uvozovány pomocí bytu *0xF0* a ukončeny *0xF7*.

Watchdog zprávy Pokud není změněna konfigurace komba jsou odesílány každých 500 ms zprávy ve formátu:

```
0xF0 0x43 0x7D 0x60 0x44 0x54 0x41 0x31 0xF7
```

Tyto zprávy se zdají být „heartbeat“ pro určení zda je zařízení připojeno. Pokud je odeslána přímo na port, na kterém poslouchá ovládací aplikace, je zobrazeno připojené kombo. Poslední byte system exclusive midi zprávy značí jeho typ. Existuje několik variant THR10 zesilovačů, každá zaměřená na jiný hudební styl. Při změně posledního bytu na hodnotu `0x32` aplikace změní vzhled a nastavení pro jiný typ komba. Tuto změnu je možné vidět na obrázku 5.3.

Zpráva interního stavu Po připojení komba k aplikaci je odesílá sekvence bytů, pomocí které se dotazuje komba na aktuální stav:

```
0xF0 0x43 0x7D 0x20 0x44 0x54 0x41 0x31 0x41 0x6C 0x6C 0x50 0xF7
```

Obrázek 5.3: THR Editor zobrazující jiný typ zesilovače.

Obrázek 5.4: THR Editor zobrazující připojené zařízení.

Poté je zaslána odpověď obsahující sekvenci několika desítek bytů. Tato odpověď udává aktuální stav hodnot v kombu. Po převedení těchto bytů na text lze vidět, že byty od pozice 18 až do 80 udávají jméno nastavení. To je poté zobrazováno v aplikaci THR Editor. Postupnou změnou jednotlivých hodnot je možné zjistit pozice, na kterých se jednotlivé hodnoty nacházejí a jejich maximální a minimální hodnoty.

Pomocí takto získaných hodnot lze nyní určit aktuální natavení komba. Tato zpráva na konci také obsahuje kontrolní součet CRC o velikosti jednoto bytu. Jedná se o poslední byt v datech midi. Pokud obdobnou zprávu, která byla přijata od komba, odešleme s rozdílnými hodnotami a korektním CRC. Tyto hodnoty budou nastaveny.

Zpráva změny Při změně libovolného nastavení komba pomocí knoflíků je odeslána zpráva informující o této změně. Ta obsahuje ID nastavení, které bylo změněno. ID jednotlivých hodnot je rozdílné od jejich pozice. Je tedy nutné si i tuto hodnotu uložit. Zpráva změny může vypadat následovně:

```
0xF0 0x43 0x7D 0x10 0x41 0x30 0x01 0x03 0x00 0x50 0xF7
```

V tomto příkladě se jedná o hodnotu s ID `0x03`, která byla změněna na novou hodnotu `0x50`.

Některé nastavené hodnoty mohou mít až dva byty. V takovém případě by hodnota zasahovala i do předchozího bytu, který je aktuálně v hodnotě `0x00`. Pokud jsou tyto zprávy odeslány do komba, dochází ke změně hodnoty s daným ID na odeslanou hodnotu.

5.6 Komunikace ovladače s mobilní aplikací

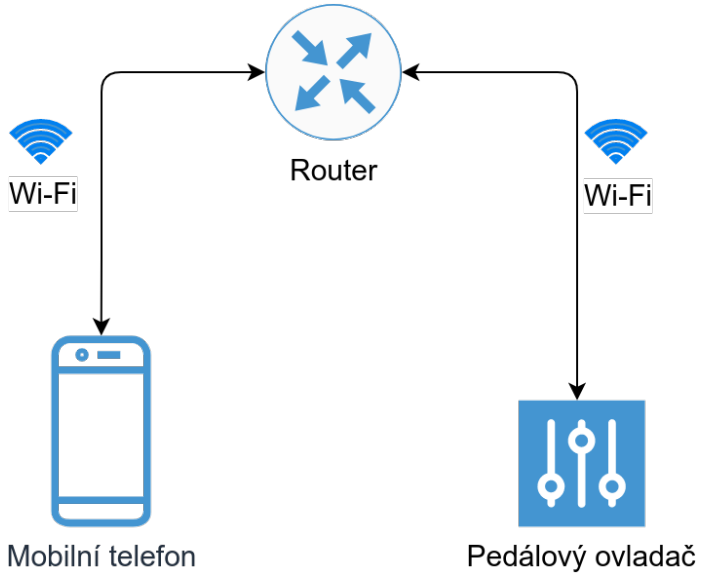
Technologie, pomocí které bude mobilní aplikace a pedálový ovladač komunikovat, musí být dostupná na většině mobilních zařízení. Je také nutné, aby samotný pedálový ovladač vybranou technologii podporoval. Tyto parametry splňují dvě možnosti.

- Bluetooth
- Komunikace po lokální síti za použití síťových socketů

Komunikace po lokální síti

Komunikace po lokální síti sebou přináší mnoho pozitiv. Jedná se o velice rozšířenou technologii, dostupnou ve většině mobilních telefonů. Je možné použít mnoho již vytvořených knihoven, které usnadní implementaci. Komunikace po lokální síti má malou odezvu a je možné přenést velké množství dat za relativně krátkou dobu. Jelikož v pedálovém ovladači se nachází *Raspberry Pi zero W*, což je model umožňující bezdrátovou komunikaci, je tato technologie pedálovým ovladačem podporována.

Problémy nastávají v dostupnosti lokálních sítí. Všechny lokální sítě nejsou veřejné a ty, které umožňují volné připojení se nenacházejí všude. Je tedy nutné, aby pedálový ovladač vytvářel vlastní lokální síť, ke které by se mobilní telefon připojil. Jelikož mobilní telefony jsou limitovány na připojení pouze do jedné sítě pomocí Wi-Fi, byl by uživatel značně limitován a při používání pedálového ovladače by nebylo možné používat Wi-Fi pro přístup k internetu. Tato komunikace je znázorněna na obrázku 5.5.

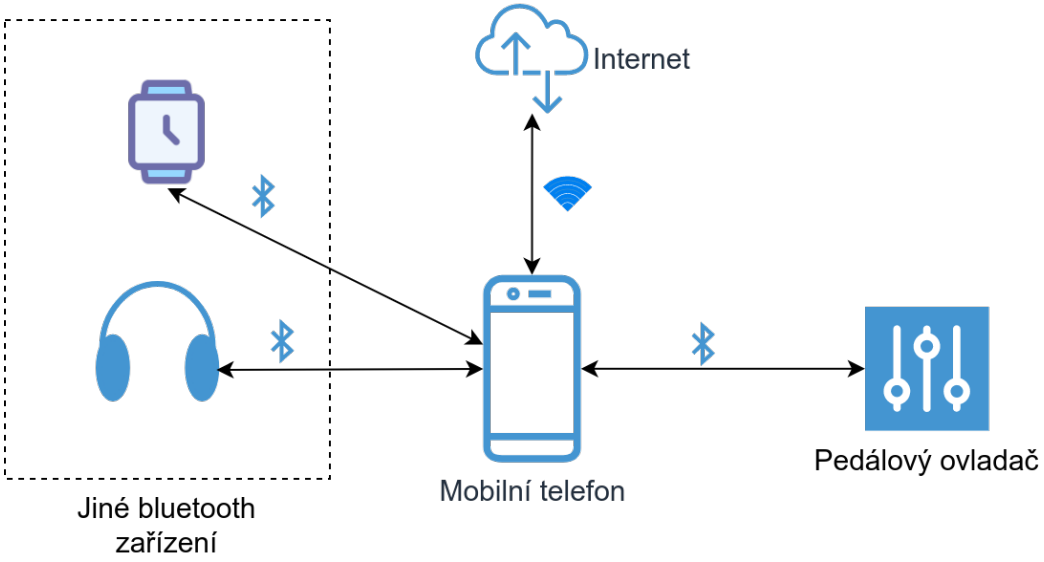


Obrázek 5.5: Komunikace po lokální sítí.

Komunikace pomocí Bluetooth

Technologii Bluetooth obsahuje většina mobilních telefonů. Umožňuje připojení k více zařízením současně a tím není uživatel omezen při používání pedálového ovladače. Technologie Bluetooth pracuje pouze na relativně malé vzdálenosti, což ale není problém, jelikož při konfiguraci ovladače je nutné ho mít při sobě pro vyzkoušení, zda nastavený tón zní přijatelně. Tento typ komunikace je znázorněn na obrázku 5.6.

Rychlost komunikace je však podstatně menší v porovnání s komunikací po lokální síti. Technologie Bluetooth je podstatně rozsáhlá a obsahuje velké množství protokolů. Analýza může tedy zabrat podstatnou dobu.



Obrázek 5.6: Komunikace pomocí bluetooth.

Výběr použité technologie

Po zhodnocení jednotlivých možností byla pro komunikaci s mobilním zařízením vybrána technologie Bluetooth, jelikož mnohem méně omezuje uživatele. Také se zdá být vhodnější pro případné budoucí rozšíření pro přenos audia.

Implementace komunikace

Jak již bylo popsáno výše, pro komunikaci bude použita technologie Bluetooth a pro implementaci knihovna Bluez. Ta je hojně používána na linuxových zařízeních a umožňuje dvě možnosti interakce. Pomocí dodávané knihovny v jazyce C, nebo skrze mezi procesovou komunikaci technologií DBUS. Při snaze o implementaci za použití knihovny v jazyce C bylo zjištěno, že některé metody, potřebné pro registraci služby do SDP, jsou označené jako *Deprecated* a není za ně dostupná náhrada. Bylo tedy použito rozhraní DBUS této knihovny, které poskytuje všechny potřebné funkce.

Komunikační protokol

Komunikační protokol s mobilní aplikací je textový, za použití notace JSON. Na počátku každé zprávy se nachází identifikátor jejího typu. Existuje několik typů komunikačních zpráv:

- `FwVersionRq` zaslání verze firmware rozšiřující desky.
- `HwStatusRq` informace o baterii a stavu rozšiřující desky.
- `PresetsRq` požadavek na všechna uživatelská nastavení.
- `CurrentPresetRq` zaslaní aktuálního nastavení z komba.
- `CurrentPresetIndexRq` požadavek indexu aktuálně vybraného uživatelského nastavení.

- `ConnectedRq` informace o připojení komba.
- `RemovePresetRq` odstranění uživatelského nastavení. Obsahuje parametr udávající index, který bude odstraněn.
- `AddPresetRq` přidání uživatelského nastavení. Obsahuje jako parametr definici uživatelského nastavení.
- `SetPresetsRq` přepíše veškerá uživatelské nastavení uložená v ovladači odeslanými hodnotami. Jako parametr je list všech uživatelských nastavení.
- `PresetSelect` výběr uživatelského nastavení s indexem předaným v parametru.
- `Lamp` nastavení světla komba. Parametr je boolovská hodnota zda je osvětlení zapnuto nebo vypnuto.
- `WideStereo` nastavení zapnutí technologie „WideStereo“. Parametr je boolovská hodnota udávající nastavovanou hodnotu.

Tyto zprávy jsou následně převedeny do protokolu ve formátu JSON a jsou odeslány na cílové zařízení. Případnou odpověď ovladač předává v zpět mobilnímu zařízení. Komunikace může vypadat následovně:

```
{"type":"Change","property":1,"value":86}
```

Serializace do formátu JSON je prováděna pomocí knihovny „kotlinx.serialization“. Ta umožňuje provádět serializaci a deserializaci přímo s objekty jazyka Kotlin. Jednotlivé funkce je možné konfigurovat pomocí anotací. Knihovna také umožňuje práci s abstrakcemi určitých typů. Mezi jednotlivými specifickými typy je potom rozlišováno pomocí položky v serializovaném formátu objektu. Tato položka je v komunikačním protokolu ovladače označena jako „type“. A udává typ zprávy a tím i datový typ souvisejícího objektu.

Problematika proudového odesílání JSON Komunikace s mobilním zařízením probíhá pomocí proudového odesílání (Stream). Při takovém odesílání je problém při použití notace JSON rozlišit, kdy končí aktuální zpráva a začíná nová. Tento problém byl vyřešen použitím oddělujícího znaku. Tento znak je „LineFeed“ s ASCII hodnotou *10*. A je tedy nutné odesílat strukturu JSON na jednom řádku. Z tohoto důvodu není možné použít formátování tzv. „Pretty print“, jelikož by takto formátovaná zpráva byla rozdělena na několik řádků a tedy obsahovala několik oddělovacích znaků.

5.7 Návrh UI mobilní aplikace

Před samotnou tvorbou uživatelského rozhraní byl vytvořen návrh, který byl následně upravován, dokud nebylo dosaženo intuitivního uživatelského rozhraní.

V návrhu na obrázku 5.7 je možné vidět nastavení jednotlivých hodnot komba pomocí ovládacích knoflíků. Ty jsou roztříděny do kategorií. Mezi kategoriemi lze přecházet gestem posunutí vpravo nebo vlevo. Název aktivní sekce je rozlišen pomocí rozdílné barvy.

V druhé části návrhu na obrázku 5.8 je možné vidět správu uživatelských nastavení. Do tohoto pohledu se uživatel dostane odsunutím konfiguračního panelu do spodní části obrazovky. Zde je možné přidávat a odebírat uživatelské nastavení a případně sledovat stav

Obrázek 5.7: Návrh nastavení komba.

Obrázek 5.8: Návrh správy uživatelského nastavení.

ovladače (Stav baterie a doba od startu). Aktivní nastavení je označeno rozdílnou barvou. Kliknutím na něj je do komba odesláno a je umožněna jeho modifikace v části konfigurace.

Při změně hodnot je uživateli zobrazena možnost uložení, kdy je nastavení odesláno do ovladače a uloženo v jeho paměti. Také je zobrazena možnost vrátit zpět předchozí hodnoty, kdy je aktuální konfigurace zahozena.

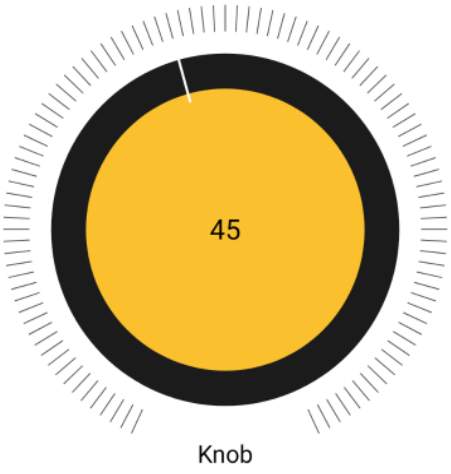
Ovládací knoflíky Použití ovládacích knoflíku namísto posuvníků pro nastavení hodnot bylo zvoleno z důvodu snahy replikovat ovládací prostředí reálného komba. Toto řešení se zdá být intuitivnější než klasické posuvníkové jezdce. Tvorba této komponenty uživatelského rozhraní je popsána v následující kapitole.

5.8 Tvorba knoflíku pro nastavení hodnot

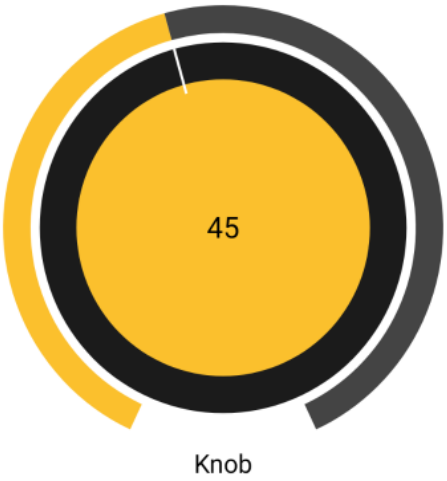
Při tvorbě vlastní grafické komponenty byl brán ohled na její znovupoužitelnost a konfigurovatelnost. Bylo potřeba vytvořit vhodnější komponentu než je klasický posuvník dostupný ve většině grafických knihoven. Tento styl byl zvolen z důvodu snahy napodobit ovládací prvky reálného komba.

Samotná komponenta je tvořena z několika základních tvarů. Vnitřního kruhu, ve kterém je zobrazována aktuální hodnota a vnějšího kruhu, obsahujícího ukazatel na stupnici. Tato stupnice je tvořena značkami ve tvaru krátkých čar. Pro případy, kdy je hodnot velké množství a vykreslení těchto čar by způsobilo snížení plynulosti aplikace, je možnost nahradit tyto značky pomocí oblouku, který barevně zobrazuje aktuální hodnotu.

Uvnitř knoflíku se zobrazuje aktuální zvolená hodnota. Ve spodní části je možné nastavit libovolný text udávající název upravované hodnoty.



Obrázek 5.9: Ovládací knoflík s čárkovým ukazatelem



Obrázek 5.10: Ovládací knoflík s obloukovým ukazatelem

Konfigurace komponenty

Grafická komponenta je tvořena způsobem, aby bylo možné ji co nejvíce konfigurovat. Jednotlivé části, ze kterých je tento ovládací prvek složen, mají nastavitelné vlastnosti.

Vnitřní a vnější kruh Konfigurovatelné parametry vnitřního a vnějšího kruhu jsou jejich barva a velikost.

Text aktuální hodnoty a název Pro text aktuální hodnoty a název je dostupné nastavení velikosti a barvy. Pro aktuální hodnotu je možné zaregistrovat modifikující funkci, která z hodnoty v číselné podobě vytvoří libovolný text. Ten je zobrazen místo aktuální hodnoty. Pro název je možné také nastavit vzdálenost od ovládacího kruhu.

Ručička ukazující na aktuální hodnotu Je možné konfigurovat barvu, délku a tloušťku ručičky ukazující na aktuální hodnotu na stupnici.

Stupnice hodnot Pokud je stupnice zobrazena pomocí čar, je možné nastavit jejich barvu, tloušťku, délku a vzdálenost od knoflíku. Pokud je nastaveno zobrazení pomocí oblouku, je možné nastavit jeho tloušťku a barvu oblouku od začátku po vybranou hodnotu a od hodnoty po konec oblouku. Lze také nastavit úhel začátku a konce stupnice.

Ovládání Ovládání komponenty má dva možné módy. Relativní, kdy je po doteku sledován úhel, který svírá pozice prvotního doteku, střed komponenty a aktuální pozice doteku. Aktuální hodnota je modifikována o tento úhel. Při absolutním ovládání je sledována pouze pozice aktuálního doteku a hodnota je vypočítána z úhlu tvořeného aktuální pozicí doteku, středu komponenty a úhlu počátku stupnice.

Vykreslení

Před vykreslením je nejprve zjištěna velikost, do které se musí komponenta vměstnat. Poté je vypočítán maximální poloměr kruhu, který může nabývat knoflík. Tato velikost je zmenšena o 20% pro vytvoření místa pro stupnici. Následně je vykreslen název ve vzdálenosti poloměru + nastavená vzdálenost. Jsou vykresleny kruhy tvořící grafiku ovládacího knoflíku.

Vykreslení stupnice záleží na nastavení. Pokud je stupnice oblouková, jsou vykresleny dva oblouky. Jeden s barvou označených hodnot a druhý s barvou stupnice. Jeden oblouk končí, kde druhý začíná a tím vzniká jeden souvislý oblouk s měnící se barvou podle nastavené hodnoty. Pokud je stupnice čárkovaná je úhel, ve kterém jsou rozloženy hodnoty vypočítán odečtením maximálního a minimálního úhlu stupnice. Následně je tento úhel rozdělen mezi počet počet hodnot (maximální hodnota - minimální hodnota). Poté jsou pro každou čárku ve stupnici vypočítány souřadnice začátku a konce, pomocí kterých je stupnice vykreslena. Výpočet je následující:

$$x_{start} = center_x + (radius_{outer} + radius_{max} * scaleDistance) * \sin(angle)$$
$$y_{start} = center_y + (radius_{outer} + radius_{max} * scaleDistance) * \cos(angle)$$

$$x_{end} = center_x + (radius_{outer} + radius_{max} * scaleDistance + radius_{max} * markerLength) * \sin(angle)$$

$$y_{end} = center_y + (radius_{outer} + radius_{max} * scaleDistance + radius_{max} * markerLength) * \cos(angle)$$

Obsluha doteku

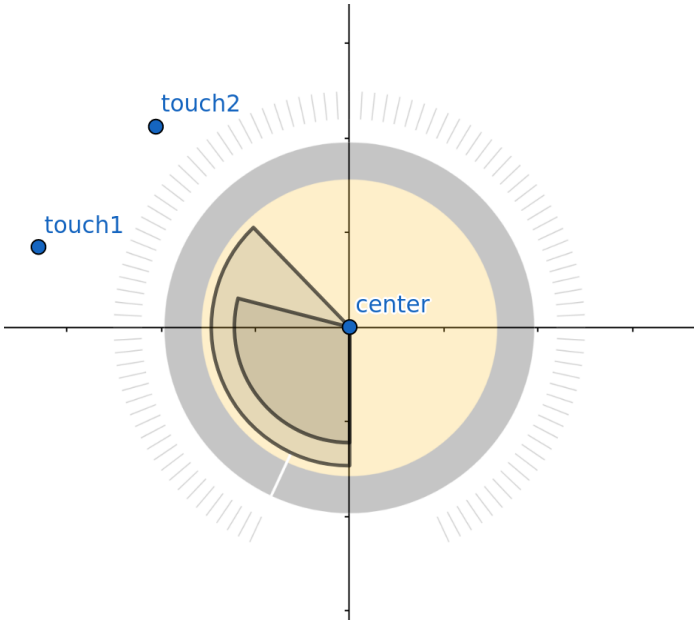
Při doteku je rozlišováno stisknutí a pohyb. Pokud je ovládací knoflík pouze stisknut, je ovládaná hodnota inkrementována. Pokud je po stisknutí taženo, nastává výpočet úhlu, o který se dotek posouvá. Tento výpočet je proveden za použití funkce *atan2* což je funkce arkus tangens, která řeší i okrajové případy dělení nulou. Svírané úhly jsou znázorněny na obrázku 5.11.

$$angleDiff = atan2(touch1_y - center_y, touch1_x - center_x) - atan2(touch2_y - center_y, touch2_x - center_x)$$

Je vypočítán úhel počátku doteku a pokud je mód ovládání nastaven na absolutní, takto vypočítaný úhel udává nastavenou hodnotu. Pokud je ovládání v relativním módu, je vypočtena změna úhlu při každém pohybu a tato změna přičtena k počátečnímu úhlu. Při uvolnění doteku je nastaven úhel na nejbližší hodnotu na stupnici.

5.9 Implementace mobilní aplikace

Mobilní aplikace byla vytvořena v prostředí Android Studio v jazyce Kotlin. Byla tvořena se snahou použít návrhový vzor MVVM.

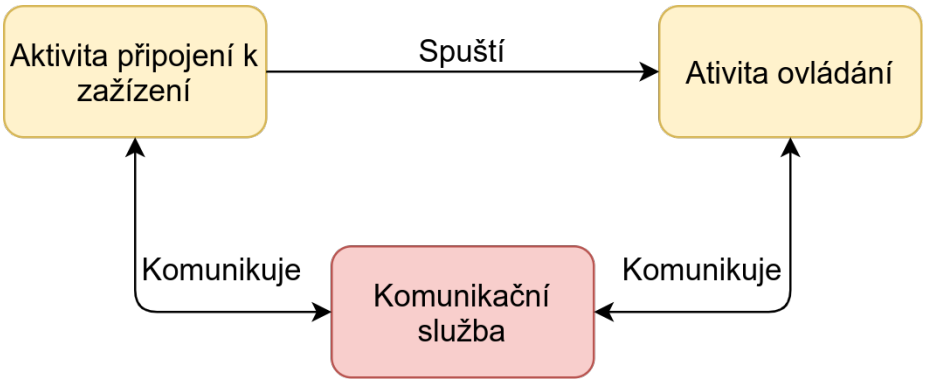


Obrázek 5.11: Výpočet úhlu při doteku

Aplikace je rozdělena do dvou aktivit. První je zobrazena po startu aplikace. Slouží k výběru zařízení, ke kterému je požadováno se připojit. Po úspěšném připojení, je spuštěna hlavní aktivita aplikace, obsahující obdobu uživatelského rozhraní popsaného v kapitole 5.7. Zde lze provádět konfiguraci komba a spravovat uživatelské nastavení.

Aplikace obsahuje službu, která běží na pozadí a stará se o obsluhu komunikace s pedálovým ovladačem. Tato služba je vytvořena při pokusu o připojení k zařízením v první aktivitě aplikace. V hlavní aktivitě je následně připojena a je použita pro veškerou komunikaci s pedálovým ovladačem.

Pokud při odesílání dojde k chybě je vyhodnoceno, že pedálový ovladač byl odpojen a uživatel je přesunut zpět na obrazovku výběru zařízení. Celková struktura aplikace je znázorněna na obrázku 5.12



Obrázek 5.12: Vnitřní struktura mobilní aplikace.

Udržování aktuálního stavu konfigurace Po spuštění hlavní aktivity aplikace je odeslán dotaz na aktuální stav komba. Po přijetí odpovědi je tento stav nastaven do View-

Modelu aplikace. Ten obsahuje hodnoty uchovávané pomocí technologie LiveData. Po nastavení nových hodnot tato technologie způsobí aktualizaci zobrazovaných dat v UI aplikace.

Pokud je změněna hodnota uživatelem za pomocí uživatelského rozhraní, je odeslána pedálovému ovladači pomocí zpráv o změně. Pedálový ovladač následně tyto zprávy konvertuje do protokolu komba a odesílá. Tyto změny jsou také prováděny ve view-modelu aplikace a tím jsou obnovovány hodnoty zobrazené v uživatelském rozhraní.

Pokud uživatel provádí změny pomocí rozhraní komba, jsou aplikací přijímány zprávy o změně. Ty jsou zpracovány změnou stavu uloženého ve view-modelu. Tím se propisují změny do uživatelského prostředí. Zprávy indikující změnu obsahují pouze identifikátor hodnoty a její aktuální stav. Tyto zprávy musí být zpracovány a podle identifikátoru změněna specifická hodnota konfigurace.

Pokud by některé zprávy byly ztraceny může vzniknout riziko, že zobrazovaný stav nebude synchronizovaný s aktuálním stavem konfigurace komba. Komunikace je prováděna pomocí Bluetooth protokolu L2CAP, který každou odeslanou zprávu potvrzuje a pokud není potvrzena je odeslána znovu. Tím ztráta odeslané zprávy není možná.

Při stisknutí tlačítka změny konfigurace na pedálovém ovladači, je aplikaci tato informace odeslána pomocí zprávy typu „PresetSelect“. Aplikace poté přepíná zobrazované nastavení na aktuální hodnotu. Uživateli je také zobrazena informace, že byla změněna konfigurace pomocí vyskakujícího textu s názvem nově zvoleného nastavení.

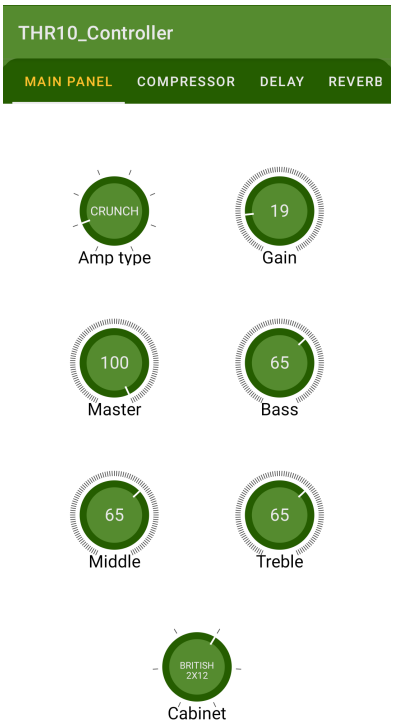
Správa uživatelských nastavení Uživateli je umožněno spravovat uživatelské nastavení. Tyto nastavení je možné měnit označením a poté konfigurací v konfigurační části aplikace. Při tvorbě nového nastavení jsou jako hodnoty použity aktuální nastavení.

Změnu pořadí je možné provést pomocí podržením na ikoně posunutí u nastavení, které je potřeba přesunout a potažním do požadované pozice. Dlouhým stiskem na je možné změnit jeho název. Smazání je provedeno stiskem ikony odpadkového koše. Při smazání není zobrazena potvrzovací hláška, namísto toho je zobrazena informace, že bylo nastavení smazáno s tlačítkem pro návrat smazaného nastavení zpět.

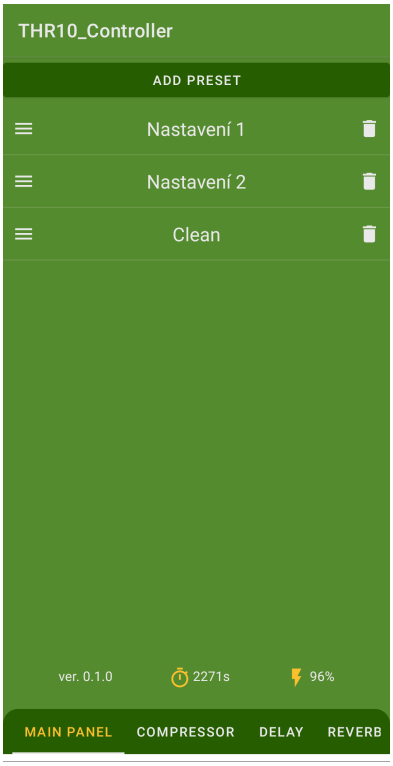
Při přidání, odebrání nebo změně pořadí se provádí synchronizace aplikace a pedálového ovladače, kdy jsou všechna nastavení mobilní aplikace odeslána do ovladače, který si je uloží do interní paměti. Výsledný vzhled části správy je možné vidět na obrázku 5.14

Konfigurační část

Konfigurační část aplikace je rozdělena podle kategorií do vlastních obrazovek. Mezi těmito obrazovkami je možné přepínat potažením vlevo/vpravo. Každá kategorie je vlastní fragment aplikace. Pokud kategorie není zobrazena, není načtena a nejsou prováděny aktualizace uživatelského rozhraní. Při zobrazení jsou načteny aktuální hodnoty z view-modelu. Tyto fragmenty jsou zobrazeny jako pojmenované záložky. Výsledný vzhled je možné vidět na obrázku 5.13



Obrázek 5.13: Konfigurační sekce aplikace



Obrázek 5.14: Správa uživatelských konfigurací

5.10 Testování systému

Aby bylo možné zajistit splnění specifikací uvedených v kapitole 4.3, je nutné provést na dokončeném systému sérii testů. Ty by také měly kontrolovat funkčnost celkového systému a zajistit aby byl systém robustní a nezhroutil se při běžném používání.

Tyto testy lze rozdělit do několika kategorií:

- *Testy funkční* testují, zda software systému odpovídá specifikacím. Dále kontrolují jeho stabilitu. Snaží se systém dostat do neočekávaných stavů, ve kterých by nemusel pracovat správně.
- *Testy hardwarové* testují specifikace hardwaru ovladače. Kontroluje se zda baterie obsažená v ovladači má dostatečnou výdrž a zda dosah bezdrátové komunikace je dostatečný. Je nutné, aby mobilní aplikace dokázala komunikovat s pedálovým ovladačem ze vzdálenosti alespoň několika metrů.

Testy funkční

Následuje popis jednotlivých testů spadajících do kategorie funkčních testů.

Test stability systému Tyto testy mají za úkol ověřit stabilitu systému při běžném používání. Měly by být testovány všechny funkce pedálového ovladače a aplikace se snahou přivodit celkový systém do neočekávaného stavu.

Test dlouhodobé stability systému Pro správnou funkčnost systému je nutné zajistit stabilitu i po dlouhé době používání. Tento test by ji měl zaručit.

Test samostatnosti pedálového ovladače Tyto testy by měly zajistit, že pedálový ovladač bude možné používat i bez připojení k mobilní aplikaci. V takovém případě nebude dostupná konfigurace, ale ovladač by měl stále umožňovat přepínání mezi nakonfigurovanými nastaveními. Ovladač musí tyto konfigurace mít uložené a i po restartu je korektně načíst a umožnit uživateli mezi nimi přepínat.

Test funkčnosti Je nutné, aby veškeré funkce dostupné uživateli pomocí mobilní aplikace nebo pedálového ovladače správně reagovaly a modifikovaly nastavení kytarového komba. Tyto testy by měly kontrolovat propojení mobilní aplikace a komba.

Test synchronizace Mobilní aplikace zobrazuje stav kytarového komba. Ten by měl odpovídat aktuálním hodnotám nastaveným v kombu. Neměla by nastat situace, kdy by dané hodnoty byly odlišné. Tyto testy by měly synchronizaci kontrolovat.

Testy hardwarové

Zde jsou popsány testy týkající se hardwarové části vyvíjeného systému

Test spotřeby Aby bylo použití pedálového ovladače praktické je nutné, aby na vestavěnou baterii vydržel co nejdelší dobu. Je třeba, aby bylo možné pedálový ovladač použít při hudebních vystoupeních. Ty trvají maximálně několik hodin.

Test dosahu Tento test kontroluje vzdálenost, ze které je možné konfigurovat pedálový ovladač. Je nutné, aby bylo možné ovladač konfigurovat ve vzdálenosti několika metrů.

Výsledky testů

Zde jsou uvedeny výsledky testování systému.

Test stability systémů Tyto testy byly prováděny propojením ovladače s mobilní aplikací a kytarového komba. Poté byla snaha v systému vytvořit neočekávanou situaci, například nečekaným odpojením mobilní aplikace. V takovém případě musí být pedálový ovladač schopen dále pracovat.

Také byla testována situace, kdy bylo neočekávaně odpojeno kytarové kombo. V tomto případě je logické, že pedálový ovladač již nebude schopný konfigurace, ale po opětovném připojení by měl obnovit svoji funkci a pracovat dle specifikací.

Tento test proběhl bez problémů a systém fungoval podle očekávání. Při odpojení mobilní aplikace bylo možné využívat všech funkcí poskytnutých pedálovým ovladačem. Při odpojení kytarového komba a jeho opětovném připojení, jej bylo nadále možné konfigurovat a byly zachovány všechny funkce systému. Test prokázal, že v konkrétních testovaných stavech dokáže spolehlivě pracovat.

Test dlouhodobé stability Tento test nebyl proveden z důvodu časové náročnosti.

Test samostatnosti pedálového ovladače Nejprve bylo nakonfigurováno několik uživatelských nastavení pomocí mobilní aplikace. Poté byla mobilní aplikace odpojena a byly testovány funkce ovladače.

Také byl testován stav, kdy rozpracovaná konfigurace nebyla odeslána a aplikace byla ukončena. V tomto případě je konfigurace nastavena v kytarovém kombu a pokud není restartováno jsou hodnoty po opětovném připojení načteny do aplikace.

Po nastavení konfigurací a odpojení mobilní aplikace si ovladač tyto konfigurace uchovává a je možné mezi nimi přepínat. Nastavení již bez mobilní aplikace není možné, ale veškerá funkcionalita ovladače je zachována. Po restartování byly znovu načteny aktuální konfigurace a opět bylo možné mezi nimi přepínat. Systém tedy splňuje požadavky tohoto testu.

Test funkčnosti Po připojení mobilní aplikace byl postupně testován každý prvek nastavitelný v mobilní aplikaci. Následně bylo kontrolováno, zda je změna propsána do komba pomocí originální aplikace dodávané ke kytarovému kombu.

Při testování byly změněny všechny hodnoty konfigurace a tyto změny byly korektně odeslány do pedálového ovladače a následně do kytarového komba. Změna pomocí kytarového komba je také korektně přeposlána do mobilní aplikace, kde je znázorněna změnou v grafickém rozhraní aplikace.

Test synchronizace Při realizaci těchto testů byla u jednotlivých hodnot provedena prudká změna vyvolaná rychlým otočením ovládacího knoflíku komba. Při tomto procesu je z komba odesláno velké množství dat a ta musí být přeposlána do mobilní aplikace. Ta je zpracuje a výsledné hodnoty musí být identické s hodnotami v kombu. Tento proces byl testován i ze strany mobilní aplikace, kdy byly provedeny prudké změny a pomocí konfiguračního programu komba byly kontrolovány zda hodnoty odpovídají.

Tyto změny byly korektně předány kytarovému kombu a výsledné hodnoty po dokončení těchto změn odpovídají těm, nastaveným v kytarovém kombu. V grafickém rozhraní je možné vidět prováděné změny a po dokončení nastavení odpovídají data z mobilní aplikace a kytarového komba. Některé změny jsou provedeny s poněkud velkou odezvou. To je způsobeno potřebou zpracovat relativně velké množství dat. Systém tedy splňuje požadavek synchronizace, ačkoliv někdy tato synchronizace může trvat déle. Obzvláště pokud je mobilní telefon ve větší vzdálenosti a spojení je méně kvalitní.

Test spotřeby V tomto testu je měřena spotřeba při různém vytížení pedálového ovladače. Následně je na základě kapacity baterie a spotřeby vypočtena teoretická doba běhu.

Jelikož rozšiřující deska pedálového ovladače umožňuje měřit odběr, je realizace testu relativně jednoduchá. Byla změřena průměrná spotřeba při různých aktivitách prováděných s pedálovým ovladačem. Po vypočtení teoretické doby běhu na baterii byla testem zkontrolována reálná hodnota.

V rámci testu spotřeby byla do softwaru pedálového ovladače dopsána funkcionalita periodického zápisu spotřeby do souboru. Takto modifikovaný software byl spuštěn a byla testována spotřeba v různých hodnotách vytížení systému. Míra vytížení byla měřena pomocí nástroje *htop*. Následně byl soubor se spotřebou zprůměrován a pomocí těchto hodnot byla vypočítána průměrná doba běhu. Tyto hodnoty jsou zaneseny do tabulky 5.1. Míra vytížení reprezentuje procentuální vytížení procesoru při měření.

Je možné vidět, že průměrná doba výdrže na baterii se pohybuje okolo 30 hodin. Dále byla testována reálná doba výdrže při minimálním vytížení. Vypozorovaná hodnota byla

Hodnota vytížení	Průměrná spotřeba	Vypočtená výdrž na baterii
100%	279.41 mA	28,631 hod
35%	252.949 mA	31,626 hod
10%	242.435 mA	32,998 hod

Tabulka 5.1: Závislost spotřeby na vytížení pedálového ovladače

přibližně 28.5 hodiny. Tyto hodnoty jsou dostatečné pro použití pedálového ovladače při hudebním vystoupení. A systém tedy splňuje požadavky tohoto testu.

Test dosahu Test byl realizován přidáním funkcionality pro konstantní komunikaci mezi pedálovým ovladačem a mobilní aplikací. Jsou odesílány data do pedálového ovladače a po příchodu jsou přeposlána zpět. Na takových datech je měřena doba od odeslání k znovu přijmutí dat. Pokud přesáhne určitou mez, komunikace se stává nepoužitelnou. Byla měřena závislost této doby na vzdálenosti mobilního telefonu od pedálového ovladače.

Při testování dosahu byly periodicky odesílány data oběma směry a byla zjišťována odezva. Vyšší odezva znamená vícenásobné odeslání (jedná se o spolehlivou komunikaci). Následně byla zvyšována vzdálenost od ovladače. Výsledná data je možné vidět v tabulce 5.2

Vzdálenost	Průměrná odezva
0m	64,75 ms
3m	59,5 ms
7m	125,36 ms
10m	195,40 ms
13m	253,30 ms
16m	472,25 ms

Tabulka 5.2: Závislost spotřeby na vytížení pedálového ovladače

Z výsledných dat je možné pozorovat zvýšení odezvy ve vzdálenosti přibližně 7 metrů. Ve vzdálenosti 10 až 13 metrů nastává toto zvýšení znovu, ale spojení bylo stále relativně stabilní. Ve vzdálenosti 16m již nebylo možné se připojit a bylo nutné spojení vytvořit blíže a následně se vzdálit. Některé pakety byly přijaty s odezvou 50ms a jiné s odezvou i přes 1000ms.

Na základě tohoto testu jsme schopni říct, že do vzdálenosti přibližně 7 metrů je spojení spolehlivé. Ve větších vzdálenostech může komunikace s pedálovým ovladačem vykazovat vyšší odezvu a synchronizace s kombem může být pomalejší. Spolehlivá vzdálenost 7 metrů je dostatečná. Uživatel, který konfiguraci provádí, ji ve většině případů chce testovat. Je tedy nutné být poblíž pedálového ovladače.

Kapitola 6

Závěr

Cílem této práce bylo navrhnout a vytvořit pedálový ovladač a mobilní aplikaci pro konfiguraci. Tento cíl byl úspěšně splněn a výsledkem je mobilní aplikace na platformu Android a pedálový ovladač založený na mikropočítači raspberry Pi zero W.

Vytvořil jsem si přehled aktuálních řešení zabývající se podobným problémem. Tyto informace jsou popsány v kapitole 2. Na základě jejich vlastností byly určeny specifikace vytvářeného řešení. Také byl vytvořen přehled o existujících mikropočítačích, které je možné použít pro realizaci tohoto problému.

Bylo nutné prostudovat literaturu zabývající se komunikací mezi hudebními zařízeními, komunikací pomocí technologie Bluetooth a pomocí síťových socketů. Tak bylo potřeba si osvojit technologie tvorby mobilních aplikací a grafického rozhraní těchto aplikací. Informace z této literatury jsou shrnuty v kapitole 3.

Na základě těchto znalostí byl rekonstruován komunikační protokol kytarového komba a jeho konfigurační aplikace. Tento protokol byl implementován do systému pedálového ovladače pro umožnění konfigurace tohoto komba. Dále byla také vytvořena mobilní aplikace, která umožňuje interakci s pedálovým ovladačem.

V rámci jeho tvorby byla také testována funkčnost a stabilita. Toto testování je popsáno v kapitole 5.10.

Pro komunikaci pedálového ovladače a komba byl použit již existující protokol. Netušil jsem, že rekonstrukce tohoto protokolu bude tak časově náročná. Naštěstí nebyl šifrovaný, jinak by tato operace byla téměř nemožná. Celkové řešení pedálového ovladače je napsáno přibližně na 3400 řádcích a mobilní aplikace přibližně na 6400 řádcích.

Budoucí rozšíření práce zahrnuje umožnění ovladače pracovat jako Bluetooth audio přijímač. Následně by přijímané audio bylo přehráváno skrze kytarové kombo a bylo by jej možné využít jako bezdrátový reproduktor. Do budoucna je také možné rozšířit pedál o funkci tzv. „Loop“, kdy lze nahrát část skladby, která bude ve smyčce přehrávána a bude sloužit jako podklad pro zbytek skladby.

Literatura

[1] *Interface Circuits for TIA-EIA-232-F*. Texas instruments, září 2002 [cit. 2021-03-12]. Dostupné z: https://www.ti.com/lit/an/slla037a/slla037a.pdf.

[2] *The Complete MIDI 1.0 Detailed Specification: Incorporating All Recommended Practices ; Document Version 96.1*. MIDI Manufacturers Association Incorporated, 2006 [cit. 2021-03-13]. ISBN 9780972883108. Dostupné z: https://books.google.cz/books?id=yna9AAAACAAJ.

[3] *Boss Katana Air: Owner's manual*. ROLAND CORPORATION, 2018 [cit. 2021-03-03]. Dostupné z: https://static.roland.com/assets/media/pdf/KTN-AIR_eng02_W.pdf.

[4] *DSL100HR & DSL40CR: Owner's manual*. Marshall Amplification, 2018 [cit. 2021-03-03]. Dostupné z: https://marshall.com/marshall-amps/products/amps/dsl/dsl40.

[5] *Using BOSS TONE STUDIO for KATANA-AIR*. ROLAND CORPORATION, 2018 [cit. 2021-03-03]. Dostupné z: https://static.roland.com/assets/media/pdf/KTN-AIR_BTS_eng03_W.pdf.

[6] *48-pin Data Sheet – megaAVR® 0-series*. Microchip Technology Inc., 2019 [cit. 2021-03-7]. Dostupné z: https://content.arduino.cc/assets/Nano-Every_processor-48-pin-Data-Sheet-megaAVR-0-series-DS40002016B.pdf.

[7] *Line 6 HX Stomp v3.0 Owner's Manual*. Yamaha Guitar Group, Inc., 2020 [cit. 2021-05-05]. Dostupné z: https://line6.com/support/manuals/hxstomp.

[8] *LiveData Overview*. Google, únor 2021 [cit. 2021-03-19]. Dostupné z: https://developer.android.com/topic/libraries/architecture/livedata.

[9] *THR30II Wireless THR10II Wireless THR10II Reference Manual*. Yamaha Corporation, duben 2021 [cit. 2021-03-03]. Dostupné z: https://cz.yamaha.com/files/download/other_assets/7/1287107/thr30ii_en_rm_b0.pdf.

[10] Bluetooth Architecture Review Board . *RFCOMM with TS 07.10* [online]. Bluetooth [cit. 2021-03-12]. Dostupné z: https://www.bluetooth.com/specifications/specs/rfcomm-1-2/.

[11] Carlsson, A., Larsson, A., Herzberg, S., McVittie, S. a Zeuthen, D. *D-Bus Specification*. freedesktop.org, 21. dubna 2020 [cit. 2021-03-14]. Dostupné z: https://dbus.freedesktop.org/doc/dbus-specification.html.

[12] Chung, J. a Metzner, J. J. Packet synchronization and identification for incremental redundancy transmission in FH-CDMA systems. In: *[1992 Proceedings] The Third IEEE International Symposium on Personal, Indoor and Mobile Radio Communications*. říjen 1992, s. 292–295 [cit. 2021-03-14]. DOI: 10.1109/PIMRC.1992.279918.

[13] Core Specification Working Group . *Bluetooth core specification* [online]. Bluetooth [cit. 2021-03-07]. Dostupné z: https://www.bluetooth.org/docman/handlers/DownloadDoc.ashx?doc_id=478726.

[14] Gryazin, E. A. Service discovery in bluetooth. *Group for Robotics and Virtual Reality. Department of Computer Science. Helsinki University of Technology, Helsinki, Finland. Published at NEC CiteSeer, Scientific Literature Digital Library*. Citeseer. 2006, [cit. 2021-03-14].

[15] Haartsen, J. C. The Bluetooth radio system. *IEEE Personal Communications*. 2000, sv. 7, č. 1, s. 28–36, [cit. 2021-03-14]. DOI: 10.1109/98.824570.

[16] Hattersley, L. *Raspberry Pi 4, 3A+, Zero W - specs, benchmarks & thermal tests* [online]. [cit. 2021-04-03]. Dostupné z: https://magpi.raspberrypi.org/articles/raspberry-pi-specs-benchmarks.

[17] Jorgensen.", H. B. B. *Beej's guide to network programming: using internet sockets*. Prosinec 2019 [cit. 2021-03-07]. ISBN 1705309909. Dostupné z: https://beej.us/guide/bgnet/pdf/bgnet_a4_c_1.pdf.

[18] Syromiatnikov, A. a Weyns, D. A Journey through the Land of Model-View-Design Patterns. In: *2014 IEEE/IFIP Conference on Software Architecture*. 2014, s. 21–30 [cit. 2021-03-14]. DOI: 10.1109/WICSA.2014.13.